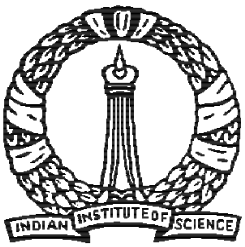
Technical Report

IISc/SID/AE/ICGEL/AOARD/2011/01

Partially Integrated Guidance and Control of UAVs for Reactive Collision Avoidance

Charu Chawla
M.S Student

Radhakant Padhi
Associate Professor



Department of Aerospace Engineering
Indian Institute of Science
Bangalore, India.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
18 OCT 2011	**Final**	**15-06-2010 to 14-06-2011**

4. TITLE AND SUBTITLE
Integrated Guidance and Control of UAVs for Reactive Collision Avoidance

5a. CONTRACT NUMBER
FA23861014014

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Radhakant Padhi

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Indian Institute of Science,Indian Institute of Science,Bangalore,India,IN,560-012

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AOARD, UNIT 45002, APO, AP, 96338-5002

10. SPONSOR/MONITOR'S ACRONYM(S)
AOARD

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
AOARD-104014

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

- Unmanned aerial vehicles (UAVs) employed in low altitude flights are liable to collide with urban structures. The present work focuses on the reactive obstacle avoidance problem for unaccountable obstacles like urban edifices, poles, etc. Unlike existing literature that mainly proposes avoidance maneuvers using kinematic and point mass models, an innovative Six-Degree of Freedom model based partial integrated guidance and control (PIGC) approach was used. PIGC performs the avoidance maneuver in the cascaded two loop structure to overcome IGC approach shortcomings, and reduces the delay in multiple loop tracking. The PIGC Six-DOF model uses nonlinear dynamic inversion technique, is computationally inexpensive, and can be implemented on onboard UAV microcomputers. A robustness study for large numbers of simulations was performed by randomly perturbing coefficients and inertia terms and clearly shows that the neuro-adaptive augmented PIGC design is more robust to parameter perturbations compared to nominal control when applied to the perturbed plant model. In all simulations, all constraints posed by the vehicle capability are very well met within the available time-to-go.

15. SUBJECT TERMS
Aircraft Control, Flight Control, UAV

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**Same as Report (SAR)**	**90**	

Table of Contents

List of Figures

Abstract

UAVs employed for low altitude jobs are more liable to collide with the urban structures on their way to the goal point. In the present work, the problem of reactive obstacle avoidance is addressed by an innovative partial integrated guidance and control (PIGC) approach using the Six-DOF model of real UAV unlike the kinematic and point mass models used in the existing literatures.

The guidance algorithm is designed which uses the collision cone approach to predict any possible collision with the obstacle and computes an alternate aiming direction for the vehicle. The aiming direction of the vehicle is the line of sight line tangent to the safety ball surrounding the obstacle. The point where the tangent touches the safety ball is the aiming point. Once the aiming point is known, the obstacle is avoided by directing the vehicle (on the principles of pursuit guidance) along the tangent to the safety ball. First, the guidance algorithm is applied successfully to the point mass model of UAV to verify the proposed collision avoidance concept. Next, PIGC approach is proposed for reactive obstacle avoidance of UAVs.

The reactive nature of the avoidance problem within the available time window demands simultaneous reaction from the guidance and control loop structures of the system i.e, in the IGC framework (executes in single loop). However, such quick maneuvers causes the faster dynamics of the system to go unstable due to inherent separation between the faster and slower dynamics. On the contrary, in the conventional design (executes in three loops), the settling time of the response of different loops will not be able to match with the stringent time-to-go window for obstacle avoidance. This causes delay in tracking in all the loops which will affects the system performance adversely and hence UAV will fail to avoid the obstacle. However, in the PIGC framework, it overcomes the disadvantage of both the IGC design and the conventional design, by introducing one more loop compared to the IGC approach and reducing a loop compared to the conventional approach, hence named as Partial IGC.

Nonlinear dynamic inversion technique based PIGC approach utilizes the faster and slower dynamics of the full nonlinear Six-DOF model of UAV and executes the avoidance maneuver in two loops. In the outer loop, the vehicle guidance strategy attempts to reorient the velocity vector of the vehicle along the aiming point within a fraction of the available

time-to-go. The orientation of the velocity vector is achieved by enforcing the angular correction in the horizontal and vertical flight path angles and enforcing turn coordination. The outer loop generates the body angular rates which are tracked as the commanded signal in the inner loop. The enforcement of the desired body rates generates the necessary control surface deflections required to stir the UAV. Control surface deflections are realized by the vehicle through the first order actuator dynamics. A controller for the first order actuator model is also proposed in order to reduce the actuator delay.

Every loop of the PIGC technique uses nonlinear dynamic inversion technique which has critical issues like sensitiveness to the modeling inaccuracies of the plant model. To make it robust against the parameter inaccuracies of the system, it is reinforced with the neuro-adaptive design in the inner loop of the PIGC design. In the NA design, weight update rule based on Lyapunov theory provides online training of the weights. To enhance fast and stable training of the weights, preflight maneuvers are proposed. Preflight maneuvers provides stabilized pre-trained weights which prevents any misapprehensions in the obstacle avoidance scenario.

Simulation studies have been carried out with the Six-DOF model of the real fixed wing UAV in the PIGC framework to test the performance of the nonlinear reactive guidance scheme. Various simulations have been executed with different number and size of the obstacles. NA augmented PIGC design is validated with different levels of uncertainties in the plant model. A comparative study in NA augmented PIGC design was performed between the pre-trained weights and zero weights as used for weight initialization in online training. Various comparative study shows that the NA augmented PIGC design is quite effective in avoiding collisions in different scenarios. Since the NDI technique involved in the PIGC design gives a closed loop solution and does not operate with iterative steps, therefore the reactive obstacle avoidance is achieved in a computationally efficient manner.

Chapter 1

Introduction

Progress of unmanned air vehicles has reduced the liability on human force both in terms of money and life expenses. It offers a fast and reliable way of accessing the information without risking the human lives or where resources utilized in human involvement are expensive than the task to be executed. Unmanned Aerial Vehicles (UAVs) have significantly been deployed in various fields like traffic monitoring, pollution control, fatal leakages, assessment of natural and man made disasters etc. Such tasks may require UAVs to fly in vicinity of urban edifices, making vehicles vulnerable for collisions. Such fatal actions may result in complete loss of information and hence mission failure. It is therefore vital that UAVs should fly autonomously to sense and avoid collisions [1]. Environment sensing is generally achieved through aboard sensors. However, the limited capability of UAVs in terms of power, size and communication interferences restricts the use of active sensing. On the contrary, passive sensing through onboard cameras has been widely implemented due to their light weight, low cost and energy usage[2].

When dangerous obstacles are sensed in the environment, UAVs must be able to react and maneuver quickly so that the collision is averted. Because of the fact that the time availability is small, the collision avoidance guidance algorithm should also be computationally efficient (preferably should be computed in closed form). To achieve this, an algorithm needs to be designed which steers the vehicle to avert the obstacle as well as plan the vehicle's path as fast enough to be implemented online. Such algorithms are called "reactive obstacle avoidance algorithms". These algorithm may sustain the challenge of heavy computational limit imposed by UAVs flying at high speeds. Apart from speed, an important requirement

is low computational resource usage, so that it may be suitable for onboard execution. Many global path planning algorithms are computationally expensive, hence they are inadequate to be solved in the limited memory usage [3]. Hence, there is a urgent need of new technique which is preferably based on sound geometric and mathematical considerations so that, due its generality and scalability, it can be applied for wide range of applications.

Path planning is the process of finding a safe flight path to the goal point. UAV path planning typically consists of two layers (i) a global path planner and (ii) a local path planner, which is primarily a reactive collision avoidance algorithm. The global planner finds a path beforehand such that known obstacles are avoided and the destination is reached. The global path is usually required to be as close to optimal as possible. When an onboard sensor detects an obstacle, the reactive collision avoidance algorithm is invoked, which computes an alternate maneuver for safely avoiding the collision.

1.1 Motivation

The problem of reactive collision avoidance for UAVs has been heavily researched in recent literature. The artificial potential field method [4] is a popular approach due to its intuitive nature and capability to be tailored to different types of problems. The potential fields in obstacle avoidance are tailored such that obstacles have a repulsive field while the destination has an attractive field. The resultant field represents a safe direction for the UAV to move along. However, this algorithm is not strictly reactive, since at every instant it takes into account the presence of all (or most of) the obstacles in the environment before deciding the direction. A model-predictive control (MPC) based collision avoidance algorithm is proposed [5], in which a potential field function is incorporated in the cost function to be minimized. The other terms in the cost function include costs for path following, control saturation and input saturation. The advantage of using MPC is that state and input constraints are accounted. The disadvantage of such a strategy is that the algorithm functions under rigid path following requirements and does not actively seek the destination. Further, MPC is a resource-intensive algorithm that requires a powerful processor, making the method unsuitable for implementation aboard a UAV. Another approach in reactive collision avoidance is RRT [3], which is a randomized search algorithm. In RRT algorithm the length of the path found is far from optimal and may have several extraneous branches due to the random

nature of the algorithm. Although reactive collision avoidance permits maneuvers that are not optimal but the wastage in the path found by RRT is significant. However, a path pruning algorithm can refine the path but such a step is infeasible for online collision avoidance. Some graph search algorithms like the best-first search algorithm are implemented for reactive collision avoidance [6]. In the best-first search method a sorted list of pre-computed motion primitives are created. Saving the pre-computing motion primitives in a lookup table is infeasible for UAV applications due to the large memory resources demanded.

Even the problem of UAV pursuing its goal is also a similar approach, which has been implemented with intermediate obstacles in the path using PN guidance based collision avoidance scheme [7]. However, this scheme leads to a jump in the control effort every time a new target is pursued. Instead, a minimum effort guidance (MEG) approach minimizes the control effort for the entire trajectory along with avoiding collisions for multiple targets [8]. A collision cone approach [9] is used to detect potential collisions by considering a threat boundary around the obstacle in MEG guidance. It has been demonstrated that MEG is more suitable than PN [8]. However, collision avoidance problems do not have minimum effort requirements and emphasize vehicle safety over low control effort. The MEG guidance causes the vehicle to maneuver until the point of impact, which is risky. Above all, the collision avoidance algorithm must ensure that the UAV full dynamics should be accounted. In some of the literature, obstacle avoidance issues are addressed through the kinematic model, [10]-[12] in which the autopilot responses are approximated by first order models. Even $3-DOF$ motion is also considered to some extent [8], [13]. This may cause vehicle to take large and practically infeasible maneuvers, leading to state or control saturation. We have observed that in many literatures the Six-DOF model, is separated into longitudinal and lateral modes [14] and these linearized modes are studied over some operating points as a linear plant. On the contrary, present work has been implemented with an innovative partial integrated guidance and control (PIGC) [18], [19] technique, which exploits full nonlinear Six-DOF model [20] of a fixed wing airplane without accounting for the decoupling modes [14].

1.2 Contribution

In the present work, a reactive obstacle avoidance algorithm is designed which has been realized in an innovative PIGC framework using full nonlinear Six-DOF model [20] of a realistic UAV. The guidance algorithm is also validated with the point mass model based formulation as a test case. The guidance algorithm detects the obstacle based on the $3D$ collision cone approach [8] and the avoidance maneuver is performed. The nonlinear guidance algorithm generates angular commands in the horizontal and the vertical plane. These commands are then pursued by the UAV to reach the aiming point [21],[22]. The aiming point is the point of contact of the tangent drawn through the UAV location to the safety ball skirting the obstacle. The tangent is the line of sight of the vehicle to the aiming point. The concept is implemented in the direction of the pursuit guidance [21] where the objective is to reorient the velocity vector of the vehicle to the line of sight (LOS) within the fraction of the available time-to-go. It finally steers UAV towards the aiming point and hence averts the obstacle. Pursuit guidance/aiming point guidance [21],[22] philosophy is used in the missile guidance to aim at the predicted position of the target at the final time.

In the present work, a relatively popular method of nonlinear control design known as nonlinear dynamic inversion technique [23], which is essentially based on the feedback linearization [24] is used in two loop cascaded structure to execute the PIGC algorithm [18], [19]. The reactive nature of the avoidance problem within the available time window demands simultaneous reaction from the guidance and control loop structures of the system i.e, in the IGC framework (executes in single loop) [17]. However, such quick maneuvers causes the faster dynamics of the system to go unstable due to inherent separation between the faster and slower dynamics. On the contrary, in the conventional design (executes in three loops)[16], the settling time of the response of different loops will not be able to match with the stringent time-to-go window for obstacle avoidance. This causes delay in tracking in all the loops which will affects the system performance adversely and hence UAV will fail to avoid the obstacle. However, the PIGC framework [18], [19], utilizes the inherent separation existing between the faster and slower dynamics of the Six-DOF model. In this way, it overcomes the disadvantage of both the IGC design [17] and the conventional design [16], by introducing one more loop compared to the IGC approach and reducing a loop compared to the conventional approach, hence named as Partial IGC.

The slower dynamics forms the outer loop and and the faster dynamics forms the inner

loop which tracks the output of the outer loop as the command for the tracking. In the outer loop, guidance command tracking is executed through nonlinear dynamic inversion (NDI) [25], where the velocity vector aligns the aiming point by enforcing the angle correction in the flight path angles, while assuring the turn coordination. The outer loop generates the body angular rates which becomes the command for the inner loop. In the inner control loop, commanded body rates are tracked in a fast dynamic inversion loop by generating the necessary control surface deflections for the vehicle. There is a separate dynamic inversion loop for velocity control which regulates the forward velocity in the body frame by generating appropriate throttle control. Moreover, the coordinated turn is implemented by ensuring zero sideslip angle through enforcement of the first order error dynamics of the side velocity in body frame to go to zero through NDI controller [23], [25]. Control surface deflections generated through inner loop and are realized by the first order actuator dynamics. The controller for the first order actuator model is designed to reduce the actuator delay and to make it robust against the parameters of the actuator model.

Nonlinear dynamic inversion [25] used in PIGC framework, is a nonlinear approach which has several advantages, like simplicity in the control structure, ease of implementation, global exponential stability of the tracking error etc [24]. The main advantage of the NDI technique is that it does not results in iterative solutions, it is instantaneously applied based on the tracking error, hence solvable online with low computational demand. However, as the NDI is rather highly sensitive to the issue of parameter inaccuracy and modeling errors, there is a strong need of augmenting this technique with some other robust/adaptive techniques [27] to make it useful in practice. A potential approach in this regard is the idea of online dynamic function approximation taking the help of evolving methods like 'neuro-adaptive technique' [28], [29]. The main philosophy that is exploited heavily in system theory applications is that neural networks have the universal function approximation property[26], which helps a controller to adapt to plants having unmodelled dynamics and time-varying parameters. Neuro-adaptive controller is designed for the inner loop to overcome the uncertainty mainly in the aerodynamic coefficients which may get amplified during inversion process. It is ensured that by applying the neuro-adaptive design based controller [29]-[31]only to the inner loop, the nonlinear and distributed uncertainties of aerodynamic coefficients in complete Six-DOF model is accounted in the closed loop. Preflight maneuvers were performed to use the stabilized weights for actual reactive obstacle avoidance to avoid any misapprehensions.

To test the performance of the PIGC scheme, different scenarios like different number and size of the obstacles in the environment have been considered and is demonstrated by using Six-DOF model [19] of a small real fixed wing UAV. The first order actuator model is also considered for the control surfaces generated in the inner loop. Various simulations have been executed with different uncertainties in the plant model to validate the robustness of the inner loop when neuro adaptive controller is reinforced on PIGC design [29], [31].

Various comparative study clearly shows that the proposed PIGC technique reinforced with neuro-adaptive controller is quite effective in avoiding collisions in different scenarios. In all the simulations, all the constraints posed by the vehicle capability are very well met within the available time-to-go. The whole task of detecting and avoiding the obstacle is based on angular correction through NDI technique which has no iterative steps. This makes the PIGC algorithm quite reactive in pursuing its objectives in a computationally efficient manner.

Chapter 2

Mathematical Model

In most of the literature, obstacle avoidance with UAVs have been executed with the kinematic model and point mass model which does not account for the actual dynamics of UAV. These approximated models invokes only the guidance of the vehicle over a specified path [10], [13]. They neglect the effect of aerodynamic forces which are observed through controllers. These models are derived with approximations on velocity vector and its directions [12]. To get the better insight of how the real UAV is controlled when commanded for a specified path, Six-DOF model is explored.

Unlike many published literature, the obstacle avoidance using PIGC algorithm proposed in this work accounts for the full nonlinear Six-DOF dynamics of the vehicle and manipulates it appropriately to avoid nearby unaccounted obstacles [18], [19]. Numerical simulations are carried out with the data of a prototype UAV, named as all electric airplane-2 ($AE-2$) [35]. It is designed and developed at the UAV lab of the Aerospace Engineering department at Indian Institute of Science, Bangalore. The $AE-2$ UAV (see Fig. 2.1) is a fixed wing airplane designed for autonomous flying with long endurance. The thrust generating unit of the $AE-2$ is an electric motor with the propeller. It has a pusher configuration for thrust generation, so that onboard sensors can be mounted at the nose. It is assumed that thrust varies linearly with the throttle input.

The mass and inertia values of $AE-2$ is given in Table 2.1.

Figure 2.1: AE-2 (Picture of All Electric airplane-2)

Table 2.1: physical data of AE-2

b	c	m	d	I_{xx}	I_{yy}	I_{zz}	I_{xz}
m	m	kg	m	kgm^2	kgm^2	kgm^2	kgm^2
2	0.3	6	0.26	0.5062	0.89	0.91	0.0015

2.1 Equations of Motion

The details of the Six-DOF dynamics of a practical UAV (see Fig. 2.1) that has been used for the simulation experiment in this work is described in this section. Under the assumptions of flat earth and airplane to be a rigid body, the complete set of Six-DOF equations of motion in body axes system are given by the following differential equations [20], [33].
Force Equations

$$\dot{U} = RV - QW - g\sin\theta + X_a + X_t \tag{2.1}$$

$$\dot{V} = PW - RU + g\sin\phi\cos\theta + Y_a \tag{2.2}$$

$$\dot{W} = QU - PV + g\cos\phi\cos\theta + Z_a \tag{2.3}$$

Moment Equations

$$\dot{P} = c_1 RQ + c_2 PQ + c_3 L_a + c_4 N_a \tag{2.4}$$
$$\dot{Q} = c_5 PR + c_6(R^2 - P^2) + c_7(M_a - M_t) \tag{2.5}$$
$$\dot{R} = c_8 PQ - c_2 RQ + c_4 L_a + c_9 N_a \tag{2.6}$$

Kinematic Equations

$$\dot{\phi} = P + Q\sin\phi\tan\theta + R\cos\phi\tan\theta \tag{2.7}$$
$$\dot{\theta} = Q\cos\phi - R\sin\phi \tag{2.8}$$
$$\dot{\psi} = Q\sin\phi\sec\theta + R\cos\phi\sec\theta \tag{2.9}$$

Navigation Equations

$$\begin{aligned}\dot{x_i} &= U\cos\theta\cos\psi + V(\sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi) + W(\cos\phi\sin\theta\cos\psi \\ &+ \sin\phi\sin\psi) \end{aligned} \tag{2.10}$$
$$\begin{aligned}\dot{y_i} &= U\cos\theta\sin\psi + V(\sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi) + W(\cos\phi\sin\theta\sin\psi \\ &- \sin\phi\cos\psi) \end{aligned} \tag{2.11}$$
$$\dot{h_i} = U\sin\theta - V\sin\phi\cos\theta - W\cos\phi\cos\theta \tag{2.12}$$

In force equations, U, V, W can be defined in terms of α and β such that

$$U = V_T\cos\alpha\cos\beta \tag{2.13}$$
$$V = V_T\sin\beta \tag{2.14}$$
$$W = V_T\sin\alpha\cos\beta \tag{2.15}$$

The definition of angle of attack (α) and side slip angle (β) are given by

$$\alpha = \tan^{-1}\left(\frac{W}{U}\right) \tag{2.16}$$
$$\beta = \sin^{-1}\left(\frac{V}{V_T}\right) \tag{2.17}$$

The coefficients $c_1 - c_9$ in Eqs. (2.4)-(2.6) are function of inertia associated with UAV in body axes system.

$$\begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_8 \\ c_9 \end{bmatrix} \triangleq \frac{1}{I_{xx}I_{yy} - I_{xz}^2} \begin{bmatrix} I_{zz}(I_{yy} - I_{zz}) - I_{xz}I_{xz} \\ I_{xz}(I_{xx} - I_{yy} + I_{zz}) \\ I_{zz} \\ I_{xz} \\ I_{xz}I_{xz} + I_{xx}(I_{xx} - I_{yy}) \\ I_{xx} \end{bmatrix} \quad \begin{bmatrix} c_5 \\ c_6 \\ c_7 \end{bmatrix} \triangleq \frac{1}{I_{yy}} \begin{bmatrix} I_{zz} - I_{xx} \\ I_{xz} \\ 1 \end{bmatrix} \tag{2.18}$$

2.2 Aerodynamic Forces and Moments

The aerodynamic force and moment coefficients are found from curve fitting on wind tunnel data of the UAV [34].The aerodynamic forces and moments are given by

$$[X_a \;\; Y_a \;\; Z_a] = \frac{\overline{q}S}{m}[-C_X \;\; C_Y \;\; -C_Z] \tag{2.19}$$

$$[L_a \;\; M_a \;\; N_a] = \overline{q}S[bC_l \;\; cC_m \;\; bC_n] \tag{2.20}$$

$$X_t = \frac{1}{m}\left(T_{max}\,\sigma_t\right) \tag{2.21}$$

$$M_t = d\left(T_{max}\,\sigma_t\right) \tag{2.22}$$

T_{max} value is $15N$ which can be produced by the electric motor and propeller assembly. σ_t is the throttle control varying from 0 to 1and d is the offset of the thrust line from the CG of the vehicle. It is assumed that thrust produced has linear relation with throttle input. Aerodynamic coefficients obtained from curve fitting on wind tunnel data [34] are given as

$$\begin{aligned} C_X &= C_{X_0} + C_{X_\alpha}(\alpha)\alpha + C_{X_{\delta_e}}(\alpha)\delta_e + C_{X_Q}(\alpha)\bar{Q} \\ C_Y &= C_{Y_\beta}(\alpha)\beta + C_{Y_{\delta_a}}(\alpha)\delta a + C_{Y_{\delta_r}}(\alpha)\delta_r + C_{Y_P}(\alpha)\bar{P} + C_{Y_R}(\alpha)\bar{R} \\ C_Z &= C_{Z_0} + C_{Z_\alpha}(\alpha)\alpha + C_{Z_\beta}\beta + C_{Z_{\delta_e}}\delta_e + C_{Z_Q}(\alpha)\bar{Q} \end{aligned}$$

$$\begin{aligned} C_l &= C_{l_\beta}(\alpha)\beta + C_{l_{\delta_a}}(\alpha)\delta_a + C_{l_P}(\alpha)\bar{P} + C_{l_R}(\alpha)\bar{R} \\ C_m &= C_{m_0} + C_{m_\alpha}(\alpha)\alpha + C_{m_\beta}(\alpha,\beta)\beta + C_{m_{\delta_e}}(\alpha)\delta_e + C_{m_Q}(\alpha)\bar{Q} \\ C_n &= C_{n_\beta}(\alpha)\beta + C_{n_{\delta_r}}(\alpha)\delta_r + C_{n_P}(\alpha)\bar{P} + C_{n_R}(\alpha)\bar{R} \end{aligned}$$

where,

$$[\bar{P} \quad \bar{Q} \quad \bar{R}] \quad = \quad \frac{1}{2V_T}[bP \quad cQ \quad bR]$$

Some of the static and dynamic derivatives are functions of α and β. Since it is a subsonic flight, mach number does not contribute to the aerodynamic derivatives. The detailed functions and constants, can be seen in [34].

2.3 Actuator Dynamics

The control deflections generated by the inner loop are passed to the actuator, modelled as a first order system. $AE-2$ [35] employs electromechanical servos for the control surface deflection, which are all similar. The actuator dynamics for the elevator servo is given by

$$\dot{\delta}_e = -9.5\delta_e + 9.5u_{\delta_e} \tag{2.23}$$

and the actuator dynamics for the throttle servo is given by

$$\dot{\sigma}_t = -4.5\sigma_t + 4.5u_{\sigma_t} \tag{2.24}$$

The actual aerodynamic control deflections observed by the vehicle are obtained through the actuator. The maximum deflection attainable from the aileron and rudder actuators is limited to $\pm 20^0$ and elevator has a lower bound of -25^0 and an upper bound of $+5^0$. A rate limit of $45^0/sec$ is applied on each of the control surface deflections.

Chapter 3

Collision Avoidance Philosophy

3.1 Collision Cone and Aiming Point Computation

Reactive collision avoidance algorithms works on the principle of sensing and avoiding the obstacle. UAV on its way to the goal point detects the unforeseen obstacle through onboard sensors and then avoid it within the available time-to-go. The "collision cone" [9] is an effective tool for (i) detecting collision and (ii) finding an alternate direction of motion that will avert the collision. In this approach, a $3D$ collision cone is constructed and analyzed for every obstacle[8]. The $3D$ collision cone approach is used to find a safe aiming point X_{ap} and the time-to-go to the aiming point t_{go}. A suitable guidance law should then be used to steer the UAV to an aiming point X_{ap} in time t_{go}. The construction of the collision cone is shown in Fig. 3.1.

A safety ball of radius r is constructed around the obstacle. In Fig. 3.1, the relative distance between the UAV and the obstacle is given by $X_r = X_{obs} - X_v$. The collision cone is formed on the plane spanned by the vectors X_r and V_T. The plane η in the safety ball which contains both the vectors forms a circle C_η. An obstacle is considered to be *critical* if the UAV is expected to violate the safety ball in the future. Therefore, future separation between the UAV and the obstacle is calculated by the projection of the X_r vector on to the velocity vector in the direction of V_T which is given as [10]

$$d_\perp = \left\| X_r - \left(\frac{X_r \cdot V_T}{\| V_T \|^2} \right) V_T \right\|$$

If $d_\perp < r$, implies the velocity vector V_T lies within the collision cone which may steer UAV towards collision. To resolve this conflict, the direction of the velocity vector, must be

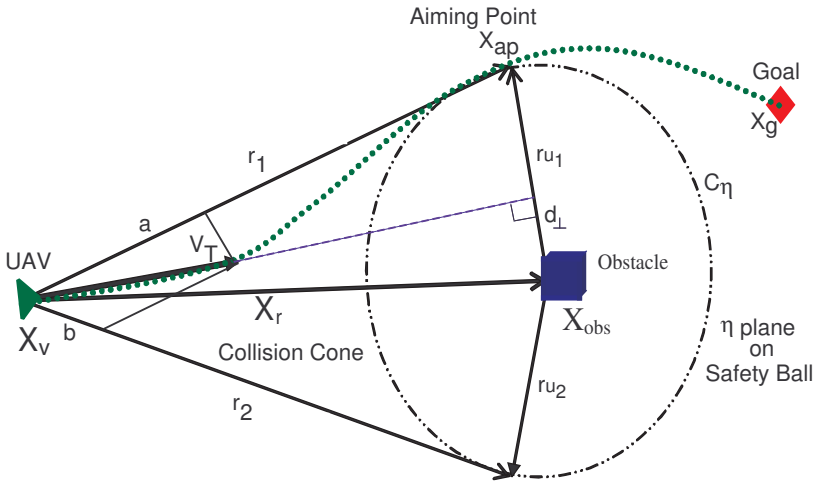


Figure 3.1: Collision cone representation

changed so that its projection line in time does not intersect the safety ball. Any movement of the relative velocity vector outside the cone will work, however the tangential solution will result in an optimal solution [11]. The optimality is achieved in terms of the minimum deviation from the path towards the goal point when obstacle is not present, hence the deviation due to the safety ball is absent. Hence, it is essentially optimal to construct the collision cone with the tangential components of the velocity vector a and b forming two tangents on the plane containing the circle C_η as shown in Fig. 3.1. The point where the tangent touches the circle η is called the aiming point. Basically, if V_T can be expressed in terms of the tangential directions r_1 and r_2 as follows:

$$V_T = ar_1 + br_2 \tag{3.1}$$

The tangential components of the velocity vector a and b are calculated as

$$a = \frac{1}{2}\left(\frac{X_r \cdot V_T}{\| X_r \|^2 - r^2} + \frac{1}{cr}\right) \tag{3.2}$$

$$b = \frac{1}{2}\left(\frac{X_r \cdot V_T}{\| X_r \|^2 - r^2} - \frac{1}{cr}\right) \tag{3.3}$$

where c is a value of scalar coefficient in Eqs. (3.2) and (3.2). The detailed derivation of a, b and c are discussed in the report [15]. The collision criterion is stated as [8]:
If $a > 0$ AND $b > 0$, the obstacle under consideration is said to be critical

To find out the aiming point so as to avert the collision when detected, the tangent vectors r_1 and r_2 are found as follows:

$$\begin{aligned} r_1 &= X_r + ru_1 \\ r_2 &= X_r + ru_2 \end{aligned} \tag{3.4}$$

Where u_1 and u_2 are the unit radius vectors of the ball perpendicular to the tangents The detailed derivation of the unit radius vectors is discussed in the report [15]. The aiming point is determined in the following way:

$$\begin{aligned} &\text{If} \quad a > b, \quad X_{ap} = X_v + r_1 \\ &\text{If} \quad b > a, \quad X_{ap} = X_v + r_2 \end{aligned} \tag{3.5}$$

The relative distance between the UAV and the aiming point can be given by $X_{vap} = X_{ap} - X_v$. The X_{vap} vector can be any one of the r_1 or r_2 tangent direction depending on the condition specified in Eq. (3.5). The time-to-go t_{go} is found as follows:

$$t_{go} = \frac{(X_{vap} \cdot V_T)}{\|V_T\|^2} \tag{3.6}$$

The problem now becomes one of guiding the UAV from $X_v(t_0) = X_{in}$ to $X_v(t_0 + t_{go}) = X_{ap}$ [8]. Note that when no obstacles are critical, the goal point becomes the aiming point, i.e., $X_{ap} = X_g$. The collision avoidance problem therefore becomes similar to a sequential target interception problem.

3.2 Nonlinear Geometric Guidance Law

The nonlinear guidance algorithm generates angular commands in the horizontal and the vertical plane. These commands are then pursued by the UAV to reach the aiming point. The aiming point is the point of contact of the tangent drawn through the UAV location to the safety ball skirting the obstacle. The tangent is the line of sight of the vehicle to the aiming point. The concept is implemented in the direction of the pursuit guidance [21] where the objective is to reorient the velocity vector of the vehicle along the line of sight to the aiming point. It finally steers UAV towards the aiming point and hence averts the obstacle. Pursuit guidance/aiming point guidance [22] philosophy is used in the missile guidance to

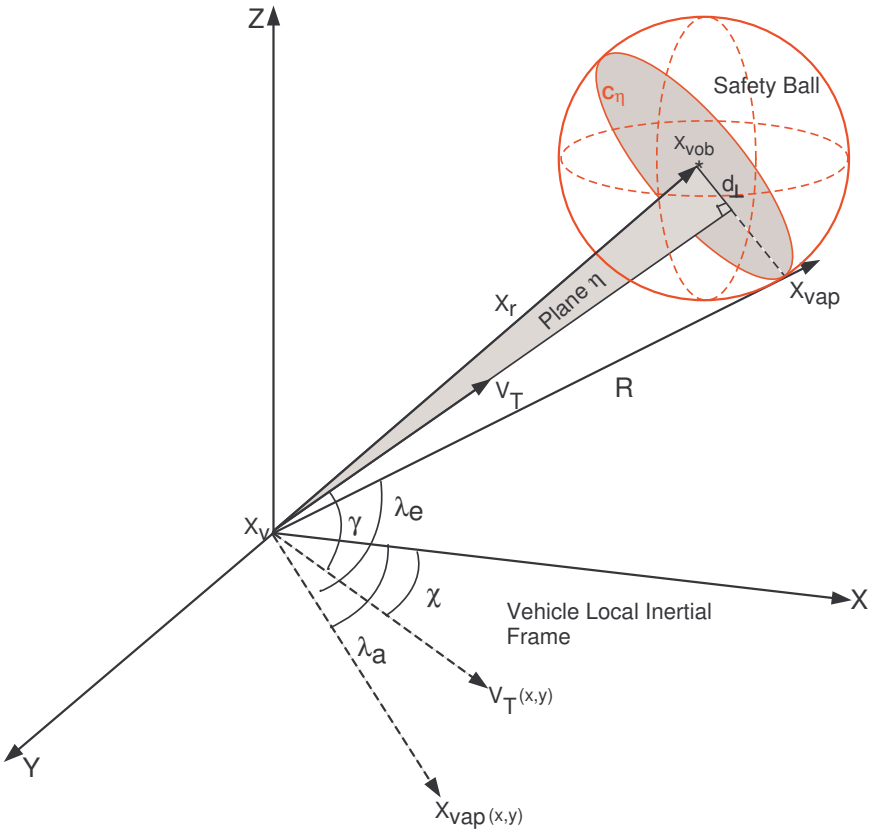


Figure 3.2: 3D view of the vectorial representation of the UAV and the aiming point for guidance logic

aim at the predicted position of the target at the final time. Figure 4.1 depicts the geometric view of the guidance logic which needs to be implemented.

In Fig. 4.1, obstacle coordinates with respect to the UAV are given by $X_{vobs} = [x_{vobs} \;\; y_{vobs} \;\; z_{vobs}]$ and the relative aiming point vector is given by $X_{vap} = [x_{vap} \;\; y_{vap} \;\; z_{vap}]$. V_T, is given by

$$V_T = \sqrt{{V_x}^2 + {V_y}^2 + {V_z}^2} \tag{3.7}$$

where $V_x = \dot{x}, \quad V_y = \dot{y}, \quad V_z = \dot{h}$. In Fig. 4.1, $V_{T(x,y)}$ vector is the projection of the velocity in the horizontal plane and similarly, vector $X_{vap(x,y)}$ is the projection of the aiming point in the horizontal plane. Similar to the case of pursuit guidance, V_T needs to be aligned to the vector pointing the aiming point i.e line of sight vector which is R distance from the vehicle, given as

$$R = \|X_{vap}\|_2 = \sqrt{x_{vap}^2 + y_{vap}^2 + z_{vap}^2} \tag{3.8}$$

The line of sight vector in $3D$ inertial space when projected in $2D$ space subtends two line of sight (LOS) angles λ_e and λ_a in vertical and horizontal plane as shown in Fig. 4.1. λ_e and λ_a definition used in the guidance logic is given by

$$\lambda_e = \tan^{-1}\left(\frac{z_{vap}}{\sqrt{x_{vap}^2 + y_{vap}^2}}\right) \tag{3.9}$$

$$\lambda_a = \tan^{-1}\left(\frac{y_{vap}}{x_{vap}}\right) \tag{3.10}$$

Similarly, in Fig. 4.1 when V_T is projected in $2D$ plane subtends two flight path angles γ and χ in vertical and horizontal plane. The expression for λ_e and λ_a in Eq.(3.9) and in Eq.(3.10) are valid provided $(x_{vap}, y_{vap}, z_{vap}) \neq (0, 0, 0)$. The objective of the aiming point guidance law is $\gamma \longrightarrow \lambda_e$ and $\chi \longrightarrow \lambda_a$ such that, at the aiming point $x_{vap} \longrightarrow 0$,$y_{vap} \longrightarrow 0$ and $z_{vap} \longrightarrow 0$ simultaneously. Equation(3.9) and Eq.(3.10) become undefined when $(x_{vap}, y_{vap}, z_{vap}) = (0, 0, 0)$. Considering the case where the denominator in Eq.(3.9) and in Eq.(3.10) becomes zero or is tending towards zero. In this case, $\lambda_e \rightarrow \pm 90^0$ depending on the sign of the numerator in Eq.(3.9) but λ_a will be undefined. To overcome the indefiniteness in Eq.(3.10) limiting value of λ_a is evaluated when $x_{vap} \rightarrow 0$ and$y_{vap} \rightarrow 0$. Limit of a function of two variables exists when no matter which direction is used to approach (x_0, y_0) (where $x_0 = 0$ and $y_0 = 0$ in the present case). To the best of our knowledge, it was found that the limit does not exist when $x_{vap} \rightarrow 0, y_{vap} \rightarrow 0$ and$z_{vap} \rightarrow 0$ because two different directions were approached which gave different values.

Therefore, the problem of undefined values for guidance commands λ_e and λ_a when $x_{vap} \rightarrow 0, y_{vap} \rightarrow 0$ and$z_{vap} \rightarrow 0$ is taken care in the present guidance problem by considering a certain bound on each of the relative coordinates $(x_{vap}, y_{vap}, z_{vap})$. It is considered that when individual coordinates $(x_{vap}, y_{vap}, z_{vap})$ will go to a very small value ε i.e. $|x_{vap}| < \varepsilon$, $|y_{vap}| < \varepsilon$ and $|z_{vap}| < \varepsilon$, where $\varepsilon = 1e - 3$ then in practical sense it is assumed that the aiming point is already achieved and the guidance command for the next aiming point is computed. Moreover, it is assumed that no further guidance command computation is required for the current aiming point. With this bound on the individual relative coordinates between the vehicle and the aiming point, it is also observed that as the aiming point is approaching nearer implies $x_{vap} \rightarrow \varepsilon, y_{vap} \rightarrow \varepsilon$ and$z_{vap} \rightarrow \varepsilon$ which causes the guidance commands in Eq.(3.9) and in Eq.(3.10) to always have a finite value.

One of the practical issue which may occur in the implementation of the collision cone approach is the violation of the safety ball after aiming point is reached.Once the UAV

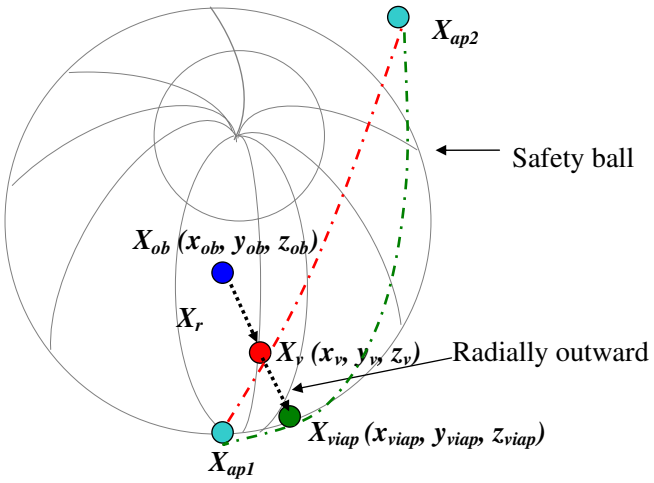


Figure 3.3: Safety ball violation after aiming point is reached

passes an aiming point, it immediately looks to maneuver towards the next aiming point. This may result in a brief violation of the first obstacle's safety bound if the direction of the new aiming point lies in the blind zone of the safety ball. Such a scenario is illustrated in Fig. 3.3. In order to solve this problem, a *sphere-tracking algorithm* is activated when $X_r < r$. The sphere tracking algorithm computes a new aiming point, called the *virtual aiming point* which is a point on the surface of the safety ball. This is found by radially extending the original relative distance line X_r until it meets the surface of the safety ball at $Xviap = [x_{viap}\ y_{viap}\ z_{viap}]$. UAV then aims for the virtual aiming point until $X_r > r$ before moving towards the next aiming point(X_{ap2}). The derivation of the virtual aiming point is elaborated in the report [15].

Chapter 4

Partial Integrated Guidance and Control (PIGC) Design

The reactive nature of the avoidance problem within the available time window demands simultaneous reaction from the guidance and control loop structures of the system i.e, in the IGC framework (executes in single loop) [17]. However, such quick maneuvers causes the faster dynamics of the system to go unstable due to inherent separation between the faster and slower dynamics. On the contrary, in the conventional design (executes in three loops)[16], the settling time of the response of different loops will not be able to match with the stringent time-to-go window for obstacle avoidance. This causes delay in tracking in all the loops which will affects the system performance adversely and hence UAV will fail to avoid the obstacle. However, the PIGC framework [18], [19], utilizes the inherent separation existing between the faster and slower dynamics of the Six-DOF model. In this way, it overcomes the disadvantage of both the IGC design [17] and the conventional design [16], by introducing one more loop compared to the IGC approach and reducing a loop compared to the conventional approach, hence named as Partial IGC.

PIGC algorithm essentially works in two loop and by using the inherent separation between the slower and the faster dynamics constituting the full nonlinear Six-DOF model [20]. The slower dynamics forms the outer loop and and the faster dynamics forms the inner loop which follows the output of the outer loop as the command for the tracking. In the outer loop, guidance command tracking is achieved when the velocity vector aligns along the LOS to the aiming point. It is enforced through the angle correction in the flight path angles,

along with the turn coordination through NDI [23]-[25]. The outer loop essentially generates the body angular rates which becomes the command for the inner loop. In the inner control loop, commanded body rates are tracked in a fast dynamic inversion loop by generating the necessary control surface deflections for the vehicle. Control surface deflections are realized by the vehicle through first order actuator dynamics. The controller for the first order actuator model is designed to reduce the actuator delay. There is a separate dynamic inversion loop for velocity control which regulates the forward velocity in the body frame by generating appropriate throttle control. Moreover, the coordinated turn is implemented by ensuring zero sideslip angle through enforcement of the first order error dynamics of the side velocity in body frame to go to zero through NDI controller.

4.1 Guidance Command Generation with Six-DOF Model

The present work assumes a goal point in the environment with unforseen obstacles whose instantaneous locations are known through onboard sensors. The obstacle locations obtained, further undergoes the necessary attitude transformation to visualize the problem with more ease in the inertial frame. With the collision cone approach [9], the aiming point is calculated with the assumption that the vehicle velocity vector V_T and the relative position vector from vehicle to the obstacle X_r spans a $2D$ plane η. The $3D$ vectorial representation of the vehicle and the aiming point in Fig. 4.1 depicts the guidance logic which needs to be implemented. In Fig. 4.1, V_T, is given by

$$V_T = \sqrt{U^2 + V^2 + W^2} \tag{4.1}$$

Referring to the section 3.2, we get the desired angles along the LOS to the aiming point as

$$\lambda_e = \tan^{-1}\left(\frac{z_{vap}}{\sqrt{x_{vap}^2 + y_{vap}^2}}\right) \tag{4.2}$$

$$\lambda_a = \tan^{-1}\left(\frac{y_{vap}}{x_{vap}}\right) \tag{4.3}$$

Similarly, the velocity vector in Fig. 4.1 when projected in $2D$ plane subtends two flight path angles γ and χ in vertical and horizontal plane respectively. The expression for λ_e and λ_a in Eq.(4.2) and in Eq.(4.3) are valid provided $(x_{vap}, y_{vap}, z_{vap}) \neq (0, 0, 0)$. The objective of the aiming point guidance law is $\gamma \longrightarrow \lambda_e$ and $\chi \longrightarrow \lambda_a$ such that, at the aiming point $x_{vap} \longrightarrow 0$,$y_{vap} \longrightarrow 0$ and $z_{vap} \longrightarrow 0$ simultaneously. Equation(4.2) and Eq.(4.3) become

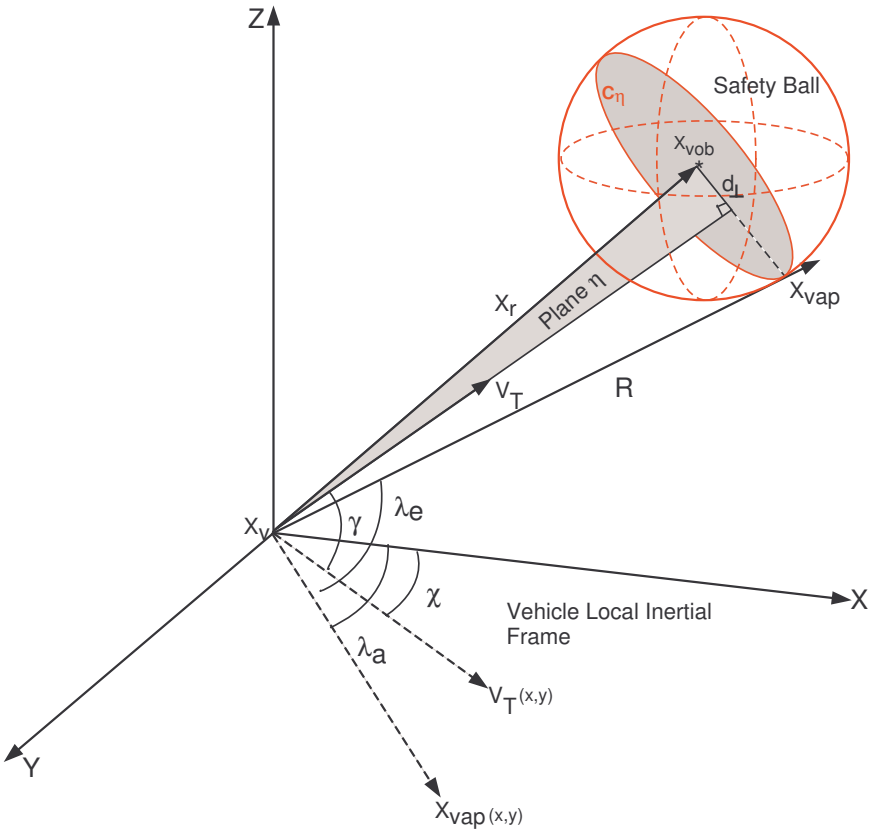


Figure 4.1: 3D view of the vectorial representation of the UAV and the aiming point for guidance logic

undefined when $(x_{vap}, y_{vap}, z_{vap}) = (0, 0, 0)$. Considering the case where the denominator in Eq.(4.2) and in Eq.(4.3) becomes zero or is tending towards zero. In this case, $\lambda_e \to \pm 90^0$ depending on the sign of the numerator in Eq.(4.2) but λ_a will be undefined. To overcome the indefiniteness in Eq.(4.3) limiting value of λ_a is evaluated when $x_{vap} \to 0$ and$y_{vap} \to 0$. Limit of a function of two variables exists when no matter which direction is used to approach (x_0, y_0) (where $x_0 = 0$ and $y_0 = 0$ in the present case). To the best of our knowledge, it was found that the limit does not exist when $x_{vap} \to 0, y_{vap} \to 0$ and$z_{vap} \to 0$ because two different directions were approached which gave different values.

Therefore, the problem of undefined values for guidance commands λ_e and λ_a when $x_{vap} \to 0, y_{vap} \to 0$ and$z_{vap} \to 0$ is taken care in the present guidance problem by considering a certain bound on each of the relative coordinates $(x_{vap}, y_{vap}, z_{vap})$. It is considered that when individual coordinates $(x_{vap}, y_{vap}, z_{vap})$ will go to a very small value ε i.e.

$|x_{vap}| < \varepsilon$, $|y_{vap}| < \varepsilon$ and $|z_{vap}| < \varepsilon$, where $\varepsilon = 1e-3$ then in practical sense it is assumed that the aiming point is already achieved and the guidance command for the next aiming point is computed. Moreover, it is assumed that no further guidance command computation is required for the current aiming point. With this bound on the individual relative coordinates between the vehicle and the aiming point, it is observed that as the aiming point is approaching nearer implies $x_{vap} \to \varepsilon, y_{vap} \to \varepsilon$ and$z_{vap} \to \varepsilon$ which causes the guidance commands in Eq.(4.2) and in Eq.(4.3) to always have a finite value.

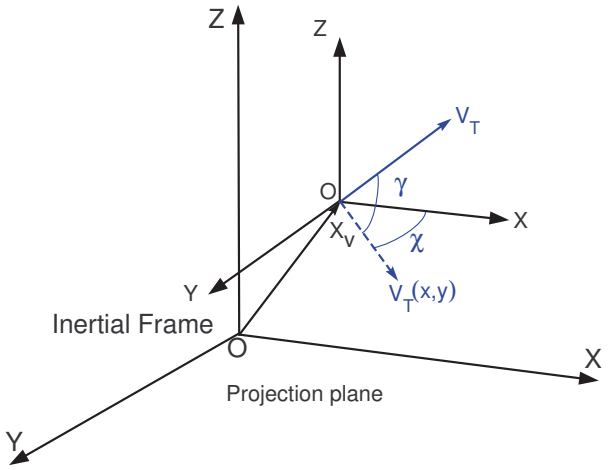


Figure 4.2: Velocity vector in the wind axes system with respect to the inertial frame

Velocity orientation angles γ and χ as shown in Fig. 4.2 can been expressed nonlinearly in terms of aerodynamic and euler angles by solving algebraically. It is performed by equating the navigation equation in the body axes system with the navigation equation in the wind axes system [20]. The navigation equations are the position rates in the inertial frame when transformed from the body axes system or the wind axes system. The navigation equations in the wind axes system with respect to the inertial frame can be derived from the Fig. 4.2. The position rates in the inertial frame are given by

$$\dot{x_i} = V_T \cos\gamma \cos\chi \tag{4.4}$$

$$\dot{y_i} = V_T \cos\gamma \sin\chi \tag{4.5}$$

$$\dot{h_i} = V_T \sin\gamma \tag{4.6}$$

The dynamics of the altitude in the inertial frame can be represented by Eq. (2.12) in

body axes and Eq. (4.6) in wind axes, therefore both the equations can be equated such as

$$V_T \sin\gamma = U\sin\theta - V\sin\phi\cos\theta - W\cos\phi\cos\theta \tag{4.7}$$

$$\sin\gamma = \frac{U}{V_T}\sin\theta - \frac{V}{V_T}\sin\phi\cos\theta - \frac{W}{V_T}\cos\phi\cos\theta \tag{4.8}$$

Equation (4.8) is simplified further by substituting Eq. (2.13), Eq. (2.14) and Eq. (2.15) such that

$$\sin\gamma = \cos\alpha\cos\beta\sin\theta - \sin\beta\sin\phi\cos\theta - \sin\alpha\cos\beta\cos\phi\cos\theta \tag{4.9}$$

Ensuring side slip angle to be zero through coordinated turn, Eq.(4.9) will become

$$\sin\gamma = \sin\theta\cos\alpha - (\sin\phi\cos\theta)\beta - \sin\alpha\cos\phi\cos\theta \tag{4.10}$$

$$\gamma = \sin^{-1}\left(\sin\theta\cos\alpha - (\sin\phi\cos\theta)\beta - \sin\alpha\cos\phi\cos\theta\right) \tag{4.11}$$

Equation (4.11) represents the nonlinear relationship between the longitudinal flight path angle γ of UAV and the body angles (aerodynamic and attitude angles) [20]. The dynamics of the vertical flight path angle can be derived from the Eq. (4.11) as follows

$$\cos\gamma\dot{\gamma} = C_1\dot{\theta} + C_2\dot{\phi} + C_3 \tag{4.12}$$

$$\dot{\gamma} = \frac{C_1\dot{\theta} + C_2\dot{\phi} + C_3}{\cos\gamma} \tag{4.13}$$

where

$$C_1 = \cos\theta\cos\alpha + (\sin\phi\sin\theta)\beta + \sin\alpha\cos\phi\sin\theta$$

$$C_2 = (\cos\phi\cos\theta)\beta + \sin\alpha\sin\phi\cos\theta$$

$$C_3 = (-\sin\theta\sin\alpha - \cos\alpha\cos\phi\cos\theta)\,\dot{\alpha} + (\sin\phi\cos\theta)\,\dot{\beta}$$

Similarly, we will obtain the nonlinear relationship between the lateral flight path angle χ and the aerodynamic and attitude angles. In the wind axes system, by dividing the navigation Eq. (4.5) by Eq. (4.4) we get

$$\frac{\dot{y}}{\dot{x}} = \tan\chi \tag{4.14}$$

Similarly, in body axes system, by dividing the navigation Eq. (2.11) by Eq. (2.10) we get

$$\frac{\dot{y}}{\dot{x}} = \left(\frac{A_1\sin\psi + \sin\beta(A_2 + A_3\cos\psi) + A_4(A_5\sin\psi - A_6\cos\psi)}{A_1\cos\psi + \sin\beta(A_2 - A_3\sin\psi) + A_4(A_5\cos\psi + A_6\sin\psi)}\right) \tag{4.15}$$

where

$$\begin{aligned} A_1 &= \cos\alpha\cos\theta\cos\beta \\ A_2 &= \sin\phi\sin\theta\sin\psi \\ A_3 &= \cos\phi \\ A_4 &= \sin\alpha\cos\beta \\ A_5 &= \cos\phi\sin\theta \\ A_6 &= \sin\phi \end{aligned}$$

It can be seen that by equating the Eq.(4.14) with the Eq.(4.15) we get

$$\tan\chi = \left(\frac{A_1\sin\psi + \sin\beta(A_2 + A_3\cos\psi) + A_4(A_5\sin\psi - A_6\cos\psi)}{A_1\cos\psi + \sin\beta(A_2 - A_3\sin\psi) + A_4(A_5\cos\psi + A_6\sin\psi)}\right) \tag{4.16}$$

Assuming $sin\alpha \simeq 0$ and $sin\beta = 0$ with due application of coordinated turn we get

$$\tan\chi \simeq \left(\frac{\cos\alpha\cos\theta\sin\psi}{\cos\alpha\cos\theta\cos\psi}\right) = \tan\psi$$

$$i.e. \quad \chi \simeq \psi \tag{4.17}$$

From Eq. (4.17), it is interpreted that the lateral flight path angle is similar to the yaw angle. With this assumption the dynamics of the horizontal flight path angle can be obtained as

$$\dot{\chi} = \dot{\psi} = Q\sin\phi\sec\theta + R\cos\phi\sec\theta \tag{4.18}$$

The objective of the guidance algorithm is to align the velocity vector along the LOS to the aiming point while ensuring a coordinated turn. In order to achieve the objective, the angle difference appeared between the relative aiming point vector X_{vap} and the velocity vector V_T in both the vertical and horizontal plane should become zero. To make the angle difference zero, both the aiming point vector and the velocity vector are projected in vertical and horizontal planes. In the vertical plane, the objective is $\gamma \to \lambda_e$ and in horizontal plane $\chi \to \lambda_a$. This implies the desired values for the orientation of the velocity vector in both the planes respectively will become

$$\gamma_d = \lambda_e, \quad \chi_d = \lambda_a$$

4.1.1 Coordinated Turn

To ensure any level turn, roll angle ϕ need not satisfy any constraint, which may cause the vehicle to skid with some side slip angle. To ensure zero side slip angle for symmetric flight, there is a need for the coordinated turn [33]. It is obtained here equivalently by demanding zero side velocity in the body frame with full control actuation. Zero side velocity is enforced through NDI formulation which generates the desired roll angle ϕ_d for the coordinated flight. The first order error dynamics corresponding to side velocity is given by

$$(\dot{V} - \dot{V}^*) + k_V(V - V^*) = 0 \tag{4.19}$$

$$\dot{V} \;\; = \;\; \dot{V}^* - k_V(V - V^*) \tag{4.20}$$

V^* is set to zero, hence $\dot{V}^* = 0$. Rearranging Eq. (2.2) gives

$$PW - RU + Y_a + gcos\theta \sin\phi_d = -k_V V \tag{4.21}$$

The desired role angle generated for coordinated turn is given as

$$\phi_d = sin^{-1}\left(\frac{-k_V V - (PW - RU + Y_a)}{gcos\theta}\right) \tag{4.22}$$

The calculation of ϕ_d observes the internal limit due to use of sine function. It implies that always $\left(\frac{-k_V V-(PW-RU+Y_a)}{gcos\theta}\right) \leq \pm 1$.

4.1.2 Outer Loop/ Guidance Command Tracking

Once the guidance commands are generated the aim is to track the guidance command such that they can be realized in terms of necessary body angular rates required by the vehicle. The outer loop tracks the guidance commands such that the aiming point is pursued while the obstacle is averted. It is slower than the inner loop due to the nature of the guidance dynamics. In the outer loop, the objective is to close in the error $E = [(\phi - \phi_d) \quad (\gamma - \gamma_d) \quad (\chi - \chi_d)]^T$ to generate necessary body angular rates $[P^* \quad Q^* \quad R^*]^T$, through first order DI controller as shown in Fig. 4.3.

The error dynamics can be represented as

$$\begin{bmatrix} \dot{\phi} - \dot{\phi}_d \\ \dot{\gamma} - \dot{\gamma}_d \\ \dot{\chi} - \dot{\chi}_d \end{bmatrix} + \begin{bmatrix} k_\phi & 0 & 0 \\ 0 & k_\gamma & 0 \\ 0 & 0 & k_\chi \end{bmatrix} \begin{bmatrix} \phi - \phi_d \\ \gamma - \gamma_d \\ \chi - \chi_d \end{bmatrix} = 0 \tag{4.23}$$

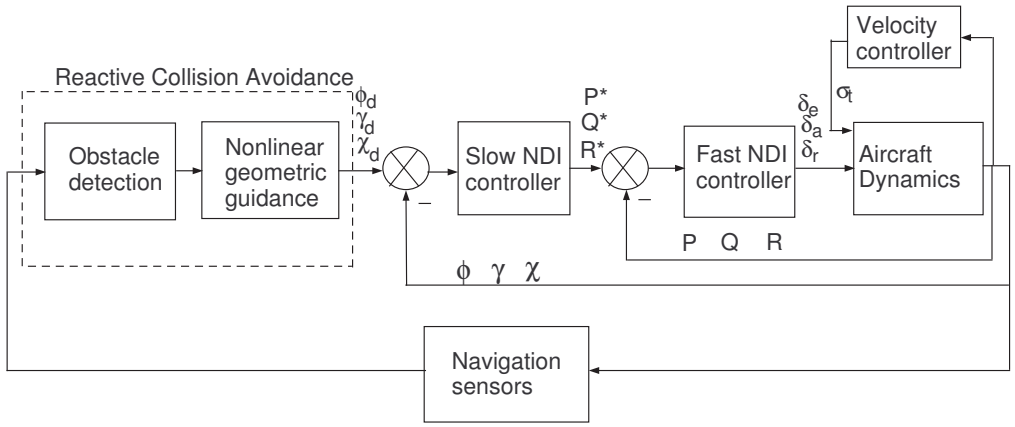


Figure 4.3: Block Diagram of PIGC design

Since the obstacle is stationary, therefore $\dot{\gamma}_d, \dot{\chi}_d$ and $\dot{\phi}_d$ are negligible and are assumed to be zero.

$$\begin{bmatrix} \dot{\phi} \\ \dot{\gamma} \\ \dot{\chi} \end{bmatrix} = \begin{bmatrix} -k_\phi(\phi - \phi_d) \\ -k_\gamma(\gamma - \gamma_d) \\ -k_\chi(\chi - \chi_d) \end{bmatrix}$$

Rearranging Eqs. (2.7), (4.13) and (4.18) such that the control of the outer loop appears in the affine form, we obtain

$$f_A + g_A \begin{bmatrix} P^* \\ Q^* \\ R^* \end{bmatrix} = b_A \tag{4.24}$$

By carrying out the necessary algebra, the closed form solution is

$$\begin{bmatrix} P^* \\ Q^* \\ R^* \end{bmatrix} = g_A^{-1}(b_A - f_A) \tag{4.25}$$

where,

$$f_A \triangleq \begin{bmatrix} 0 \\ C_3 \sec\gamma \\ 0 \end{bmatrix}$$

$$g_A \triangleq \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ C_2\sec\gamma & \sec\gamma(C_1\cos\phi + C_2\sin\phi\tan\theta) & \sec\gamma(C_2\cos\phi + \tan\theta - C_2\sin\phi) \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix}$$

$$b_A \triangleq \begin{bmatrix} -k_\phi(\phi - \phi_d) \\ -k_\gamma(\gamma - \gamma_d) \\ -k_\chi(\chi - \chi_d) \end{bmatrix}$$

The outer loop transforms the guidance commands into the desired angular rates for the inner loop. The inner loop generates the necessary control surfaces required for tacking the desired angular rates as shown in Fig. 4.3.

4.2 Inner Loop Control Design

The inner loop generates the necessary control surfaces to meet the objective of averting the obstacle and reaching the goal point efficiently. The throttle control is also generated to control the thrust input which will keep the forward velocity constant.

4.2.1 Body Angular Rate Control

To avoid the obstacle quickly and optimally it is required that the body angular rates should track the desired body rates generated by the outer guidance loop. We can write the objective as $[P \quad Q \quad R]^T \longrightarrow [P^* \quad Q^* \quad R^*]^T$. Let error be $e = [(P - P^*) \quad (Q - Q^*) \quad (R - R^*)]^T$. Writing first order error dynamics

$$\begin{bmatrix} \dot{P} - \dot{P}^* \\ \dot{Q} - \dot{Q}^* \\ \dot{R} - \dot{R}^* \end{bmatrix} + \begin{bmatrix} k_P & 0 & 0 \\ 0 & k_Q & 0 \\ 0 & 0 & k_R \end{bmatrix} \begin{bmatrix} P - P^* \\ Q - Q^* \\ R - R^* \end{bmatrix} = 0 \tag{4.26}$$

$$\begin{bmatrix} \dot{P} \\ \dot{Q} \\ \dot{R} \end{bmatrix} = \begin{bmatrix} \dot{P}^* - k_P(P - P^*) \\ \dot{Q}^* - k_Q(Q - Q^*) \\ \dot{R}^* - k_R(R - R^*) \end{bmatrix}$$

Under the assumption $\dot{P}^* = 0, \dot{Q}^* = 0$ and $\dot{R}^* = 0$, the state and the control terms are separated. Rearranging the moment equations $\dot{P}$, $\dot{Q}$, $\dot{R}$ in Eqs. (2.4), (2.5) and (2.6), we get

$$f_R + g_R U_c = b_R \tag{4.27}$$

By carrying out the necessary algebra, the closed form solution obtained by inverting the body rates dynamics is

$$U_c = g_R^{-1}(b_R - f_R) \tag{4.28}$$

where, $U_c = [\delta a \;\; \delta e \;\; \delta r]^T$ and other terms are defined as follows

$$f_R \triangleq \begin{bmatrix} c_1 RQ + c_2 PQ + c_3 L_{a_x} + c_4 N_{a_x} \\ c_5 PR + c_6(P^2 - R^2) + c_7(M_{a_x} - M_t) \\ c_8 PQ - c_2 RQ + c_4 L_{a_x} + c_9 N_{a_x} \end{bmatrix}$$

$$g_R \triangleq \begin{bmatrix} c_3 L_{a_u} & 0 & c_4 N_{a_u} \\ 0 & c_7 M_{a_u} & 0 \\ c_4 L_{a_u} & 0 & c_9 N_{a_u} \end{bmatrix}$$

$$b_R \triangleq \begin{bmatrix} \dot{P}^* - k_P(P - P^*) \\ \dot{Q}^* - k_Q(Q - Q^*) \\ \dot{R}^* - k_R(R - R^*) \end{bmatrix}$$

where,

$$\begin{aligned} L_{a_x} &\triangleq \bar{q}Sb[C_{l_\beta}(\alpha)\ \beta + C_{l_P}(\alpha)\ P + C_{l_R}(\alpha)\ R] \\ M_{a_x} &\triangleq \bar{q}Sc[C_{m_0} + C_{m_\alpha}(\alpha)\ \alpha + C_{m_\beta}(\alpha, \beta)\ \beta + C_{m_Q}(\alpha)\ Q] \\ N_{a_x} &\triangleq \bar{q}Sb[C_{n_\beta}(\alpha)\ \beta + C_{n_P}(\alpha)\ P + C_{n_R}(\alpha)\ R] \end{aligned}$$

and,

$L_{a_u} \triangleq \bar{q}SbC_{l_{\delta a}}$; $M_{a_u} \triangleq \bar{q}ScC_{m_{\delta e}}$; $N_{a_u} \triangleq \bar{q}SbC_{n_{\delta r}}$

The gain selection in the inner loop is done judiciously due to the faster dynamics of the inner loop than the outer loop. This leads to low settling time of the inner loop dynamics compared to higher settling time of the outer loop.

4.2.2 Velocity Control

The forward velocity is maintained constant in the steady level flight. It is achieved by controlling the thrust through throttle control. Here, thrust is assumed to vary linearly with the throttle control. We can write error in forward velocity in body frame as $(U - U^*)$, where U^* is the desired value, and it is the initial trim value of the forward velocity. Enforcing the first order error dynamics.

$$(\dot{U} - \dot{U}^*) + k_U(U - U^*) = 0 \tag{4.29}$$

$$\dot{U} = \dot{U}^* - k_U(U - U^*) \tag{4.30}$$

Separating the state and control terms in $\dot{U}$ from Eq. (4.30) and rearranging

$$f_U + g_U \sigma_t = b_U \tag{4.31}$$

The throttle control obtained in closed form is

$$\sigma_t = g_U^{-1}(b_U - f_U) \tag{4.32}$$

where,

$$\begin{aligned} f_U &\triangleq RV - QW - gsin\theta + X_a \\ g_U &\triangleq \frac{T_{max}}{m} \\ b_U &\triangleq \dot{U}^* - k_U(U - U^*) \end{aligned}$$

4.2.3 Actuator Controller

The control surfaces generated from the inner loop dynamics are passed through the first order actuator model before it is given as a control to the plant dynamics. It is a open loop mode where the tracking error introduced by the actuator model due to the first order delay is cumulative which may adversely affects the system performance. Therefore, in order to compensate for the lag either a fast actuator should be used or a controller for the actuator should be designed based on the tracking error. In the present study, we prefer to design a controller for the first order actuator with the assumption that the actuator states (control deflections) are available for the feedback. The controller is designed based on the error of the actual states of the actuator $\sigma_t,\ \delta_e,\ \delta_a,\ \delta_r$ and the desired state of the actuator $\sigma_t^*,\ \delta_e^*,\ \delta_a^*,\ \delta_r^*$ respectively. Enforcing the first order error dynamics on the error $e = \delta_e - \delta_e^*$ of the elevator deflection we get

$$(\dot{\delta_e} - \dot{\delta_e^*}) + k_{\delta_e}(\delta_e - \delta_e^*) = 0 \tag{4.33}$$

$$\dot{\delta_e} \;=\; \dot{\delta_e^*} - k_{\delta_e}(\delta_e - \delta_e^*) \tag{4.34}$$

Substituting the actuator dynamics from the Eq.(2.23) in Eq. (4.34) and rearranging we get

$$u_{\delta_e} = \frac{1}{9.5}[-9.5\delta_e + \dot{\delta_e^*} - k_{\delta_e}(\delta_e - \delta_e^*)] \tag{4.35}$$

Similar controller is designed for all the control surfaces such that the settling time of the throttle controller is higher than the elevator, aileron and rudder controllers. The actuator dynamics observes the rate limits and the output of the actuator observes the position limits. This mode of operation of the actuator is called the closed loop mode of the actuator due to the feedback of the actuator states. The feedback controller for the actuator has more tolerance against the inaccurate parameters of the actuator model compared to the actuator in the open loop mode and also overcomes the first order delay introduced by it.

4.3 Numerical Results

For the present study, to validate the reactive nonlinear guidance algorithm, different scenarios were considered. For simulations, full, nonlinear Six-DOF model of UAV is integrated through Runge-Kutta method for better accuracy [36]. The time- to-go required to avoid each obstacle is around $4sec - 8sec$. The gain selection corresponding to the outer loop and the inner loop is dictated by the settling time of the system dynamics and required time-to-go to reach the aiming point. Obstacles are surrounded with the safety ball whose radius varies from $5m - 20m$. The choice of the radius of the safety ball depends on the location of the obstacles in the environment. The obstacles with different number and radius of the safety ball are considered. It is implemented because in practice, the obstacles of different sizes will be encountered, which should be sensed with appropriate safety ball size around them.

In all the scenarios, the experiment were conducted with the actuator model, both in the open loop and closed loop mode. In the open loop mode, the generated control surface deflections are passed through the first order actuator model so as to obtain actual control deflections observed by the vehicle. Actuator in open loop mode generates large tracking error which is compensated by operating the actuator in the closed loop. In the closed loop mode, a controller is designed for the actuator state (control surface deflections) which facilitates in reducing the tracking error. Both the open loop and closed loop mode operation of actuator observes the rate limit and position limit as posed by the system.

Moreover, algorithm is also validated for large number of simulations. Simulation study includes various cases like perturbed initial conditions of the states of the vehicle, varied size and position of the multiple obstacles on the path to the goal point.

4.3.1 Trim Conditions

Trim conditions are calculated for steady level flight at a given velocity and altitude by equating the dynamic equations to zero and then solving numerically [20][34]. The state vector $X = [U,\ V,\ W,\ P,\ Q,\ R,\ \phi,\ \theta,\ \psi,\ x_i,\ y_i,\ h_i]$ representing the UAV is initialized with trim values. The trim values used for the current problem are shown in Table 4.1. The

Table 4.1: Trim values of the state and control variables

Velocity of UAV	V_T=20 m/sec
Position coordinates	$x_{trim} = 0m,\ y_{trim} = 0m,\ h_{trim} = 50m$
Body rates	$P_{trim} = 0deg/sec,\ Q_{trim} = 0deg/sec,\ R_{trim} = 0deg/sec$
Euler angles	$\phi_{trim} = 0^o,\ \theta_{trim} = 3.1339^o,\ \psi_{trim} = 0^o$
Aerodynamic angles	$\alpha_{trim} = 3.1339^o,\ \beta_{trim} = 0^o$
Control surface deflections	$\sigma_{trim} = 0.3708,\ \delta_{e_{trim}} = -3.2673^o,\ \delta_{a_{trim}} = 0^o,\ \delta_{r_{trim}} = 0^o$

following constraints were imposed by the actuator on the control variables associated with the UAV as shown in Table 4.2.

Table 4.2: Actuator constraints description (N/A: Not Applicable)

	$Engine(throttle)$	$Elevator(\delta_e)$	$Aileron(\delta_a)$	$rudder(\delta_r)$	ϕ_d
upper level limit	1	+5 deg	+15 deg	+15 deg	+45 deg
lower level limit	0	-25 deg	-15 deg	-15 deg	-45 deg
upper rate limit	N/A	+45 deg/sec	+45 deg/sec	+45 deg/sec	N/A
lower rate limit	N/A	-45 deg/sec	-45 deg/sec	-45 deg/sec	N/A

It is assumed that the instantaneous position of the obstacles are obtained through the aboard passive sensors [2]. Since the attitude of the vehicle is changing so the sensor direction will change and it may cause the motion of the static view in its frame. Keeping in view, the delicacy of the sensor frame the obstacle locations are assumed to undergone the attitude transformation so as to perform the guidance command computation with ease in the inertial frame. Two scenarios are considered, one with the single obstacle and other with the multiple obstacles. In all the scenarios, the effect of the actuator and the vehicle constraints are taken into account.

4.3.2 Scenario 1: Single Obstacle

In case of single obstacle, the position of the obstacle in inertial frame after attitude transformation is $X_{obs} = [100 \quad -10 \quad 48]$. The starting point of the UAV where it senses the obstacle is $X_i = [0 \quad 0 \quad 50]$ and the goal point is $X_g = [300 \quad -20 \quad 45]$. In the results, where command tracking is required, the responses only with the actuator in open loop mode are represented. The results with no actuator effect are validated and hence are not included for the clarity of the results in case of command tracking. The actuator in open loop mode is considered for the present case study.

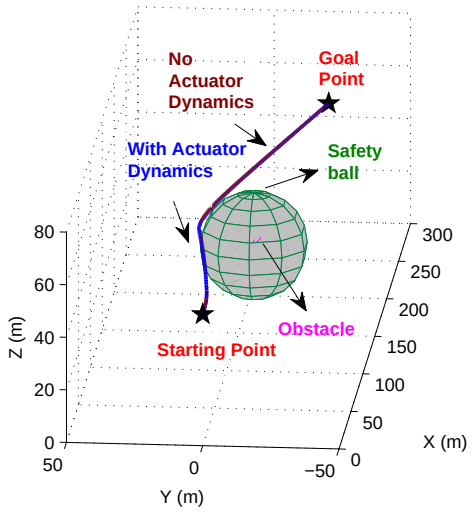


Figure 4.4: 3D view of the trajectory for obstacle avoidance

Figure 4.4 shows the effectiveness of the guidance algorithm in avoiding the static obstacle and finally reaching the goal point. Two cases are shown in Fig. 4.4, the blue trajectory represents the case with the actuator effect included into the system and the brown trajectory represents the case in which, the controls generated are directly given into the system. The actuator effect introduces a delay in the control response compared to the case with no actuator. The trajectories with and without actuator model are plotted with are not distinguishable in case of single obstacle avoidance scenario as shown in Fig. 4.5. Figure 4.5 shows the $2D$ view of the avoidance scenario, in X-Y plane. Figure 4.6 represents the longitudinal control effort required by the vehicle and represents three comparative control profiles. The reference command in the Fig. 4.6 represents the input to the actuator (i.e the control generated from the inner loop) and the output of the actuator in open loop mode (i.e

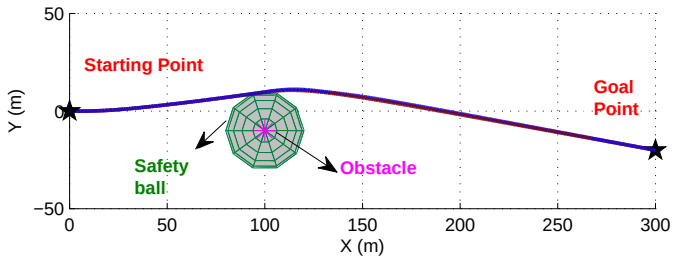


Figure 4.5: 2D view of the trajectory for obstacle avoidance in X-Y plane

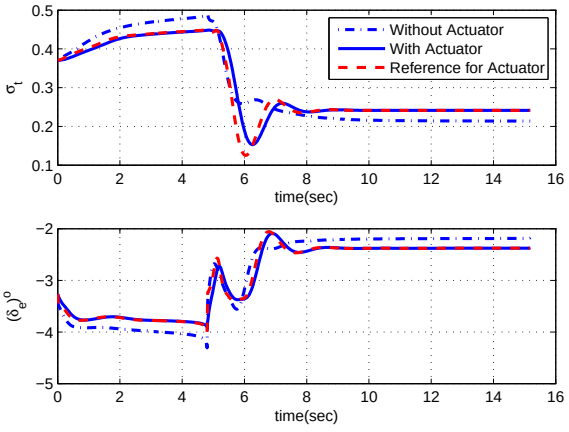


Figure 4.6: Longitudinal control surface deflections

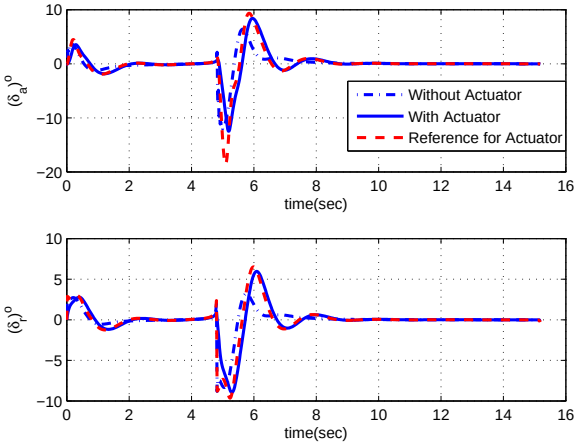


Figure 4.7: Lateral control surface deflections

the control with first order delay)is tracking the reference value. The third control profile represents the case when actuator effect is not considered in the system. In Fig. 4.6, throttle control increases to keep the forward velocity $U = constant$. It is due to the fact that the velocity will tend to decrease due to drag being more than the total lift. It happens during turning that only one component of the lift vector is available to balance the weight of the UAV. Figure 4.7 shows the response of the lateral control effort required by the vehicle. Figure 4.7 represents three comparative control profiles with the actuator output in open loop mode following its reference value (i.e input to the actuator) and the case of without actuator effect in the system. The aileron and rudder control with respect to the avoidance maneuver are within their bounds. It can be seen that both the deflections settles down to their trim values when avoidance is over.

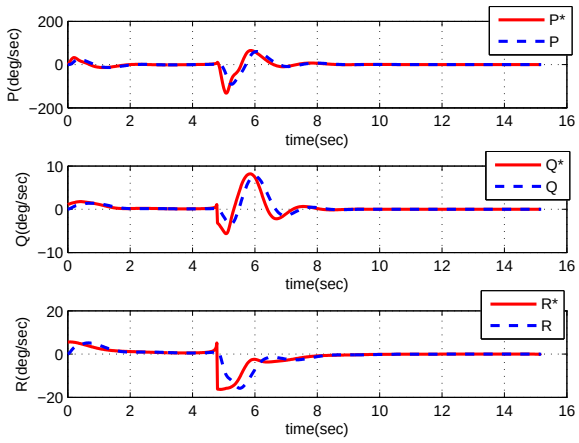


Figure 4.8: Tracking of commanded body angular rates

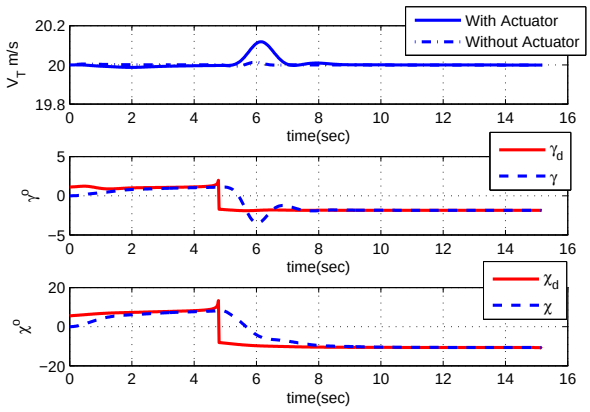


Figure 4.9: Total velocity and its direction

Figure 4.8, represents the command tracking by considering only the case of the actuator in the loop. It can be seen that the judicious use of the gain selection of the inner loop causes the body angular rates to track their commanded values efficiently. To maintain the clarity of the results, Fig. 4.8 the response without actuator has been verified and is more satisfactory than the actuator in the open loop mode. In Fig. 4.9 it can be seen that in the outer loop, flight path angles γ and χ tracks the guidance command in presence of the actuator in open loop mode which shows the successful achievement of the aiming point and the goal point. It can be inferred from the Fig. 4.9 that the total velocity (V_T) of the UAV remains almost constant and deviates more in case of the actuator in open loop, when the

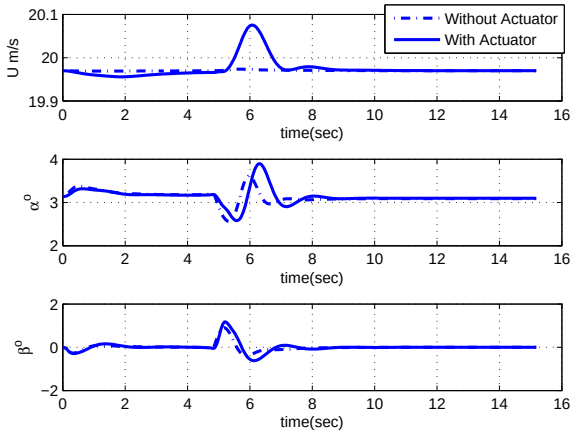


Figure 4.10: Forward velocity and aerodynamic angles

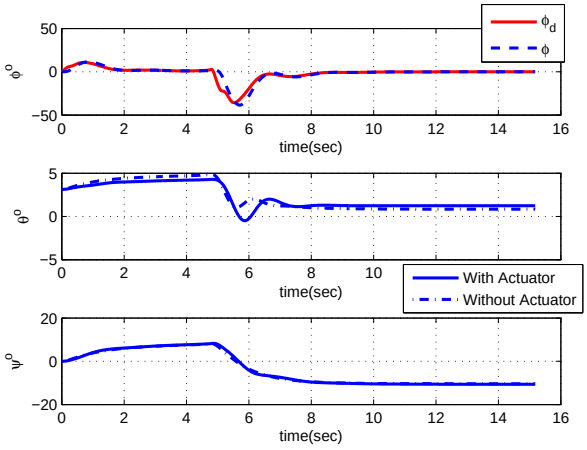


Figure 4.11: Tracking of roll angle and euler angles

obstacle is avoided.

Figure 4.10 shows the forward velocity profile and the aerodynamic angles with and without actuator effect taken into account. The aerodynamic angles α and β both with and without actuator effect as shown in Fig. 4.10 gets perturbed only when the obstacle avoidance takes place. Figure 4.11 shows the euler angles response which represents the attitude of the UAV. It can be seen from the Fig. 4.11 that the roll angle tracks its commanded value quickly even with actuator in the loop without violating the turning constraint. It can be inferred from Fig. 4.11 that the attitude defining angles settles down to their steady state values both in case of with and without actuator dynamics, when the obstacle is averted.

4.3.3 Scenario 2 : Multiple Obstacles

To avert the collision, a minimum separation distance of $50m$ is required between the obstacles. It has also been observed through simulations that the separation between the UAV and the obstacle should be at least five times the radius (r) of the ball with which the obstacle is surrounded. These constraints on the guidance algorithm are imposed by the vehicle capability considered for the current problem. In case of the multiple obstacles, two cases have been considered, one with two obstacles and other with three obstacles. In all the scenarios the effect of the first order actuator model in open loop mode has been carried out. In the results, where command tracking is required, the responses only with the actuator in

open loop are represented. The results with no actuator effect are validated and hence are not included for the clarity of the results in case of command tracking. In both the cases, the starting point of the UAV is same as $X_i = [0 \quad 0 \quad 50]$. Results for both the cases have been discussed together for better clarity.

- Case1: The position of the obstacles in the inertial frame are $X_{obs1} = [100 \quad -10 \quad 48]$, $X_{obs2} = [250 \quad 5 \quad 50]$ and the goal point is $X_g = [500 \quad -25 \quad 60]$.
- Case2: The position of the obstacles in the inertial frame are $X_{obs1} = [100 \quad -5 \quad 48]$, $X_{obs2} = [210 \quad 20 \quad 50]$ and $X_{obs3} = [300 \quad 10 \quad 48]$. The goal point which needs to be reached is $X_g = [500 \quad 5 \quad 48]$.

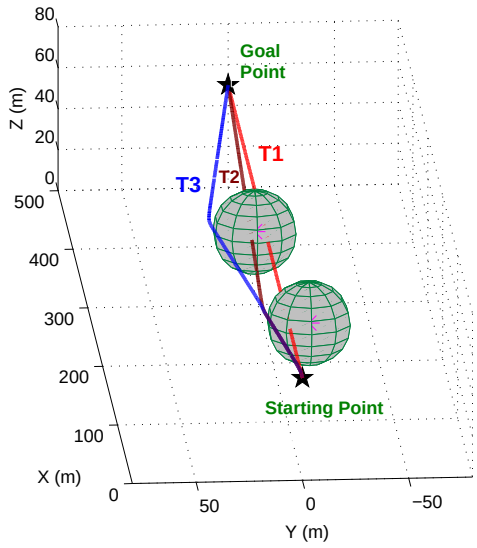


Figure 4.12: 3D view of the trajectories for obstacle avoidance

Figure 4.12 shows three different trajectories of the UAV with the actuator model effect taken into account. The trajectory T_1 represents the case where no guidance algorithm is invoked and the obstacle is not detected. The vehicle reaches the goal point without avoiding the obstacle. The trajectory T_2 represents the case of avoiding the single obstacle on the way to the destination point. It can be seen in Fig. 4.12 that the guidance algorithm is efficient in avoiding the single obstacle. The trajectory T_3 represents the case where one more obstacle is added to the scenario. It can be inferred that the irrespective of the number of the obstacles, the obstacle avoidance algorithm works. It can be seen in Fig. 4.13 that the

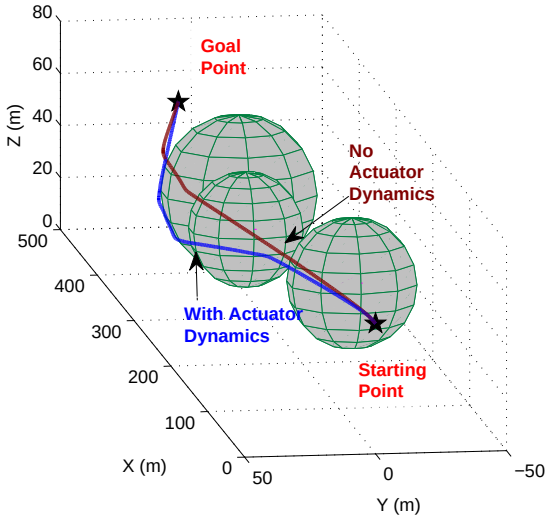


Figure 4.13: 3D view of the trajectory for obstacle avoidance

guidance algorithm performs well even with increasing number of obstacles with different size of safety ball around them. This shows the capability of the algorithm to work in the cluttered environment. It can be seen in Fig. 4.13 the two different avoidance path which corresponds to the presence and absence of the actuator in the system. The blue colored trajectory represents the case with the actuator model in open loop and the brown(dotted) trajectory represents the case without the actuator as shown in Fig. 4.13. Figure 4.14 and Fig. 4.15, gives a better view in the 2D plan of the optimal avoidance maneuver executed by the UAV in the scenario and finally reaching the goal point in both the cases - with and without the actuator model.

Figure 4.16 and Fig. 4.17 represents three longitudinal control profiles. The throttle value controls the thrust input to maintain constant forward velocity. In Fig. 4.16 the control effort is smooth due to the presence of the actuator. In Fig. 4.17, it can be seen that the response with the actuator in the loop causes high demand in throttle. In both the figs.4.16 and 4.17, the actuator output follow its reference value (i.e input to the actuator). Even in Fig. 4.18 and Fig. 4.19 three lateral control profile are shown where the aileron and the rudder deflections obtained from the actuator in open loop follows their reference value (i.e input to the actuator) with first order delay.

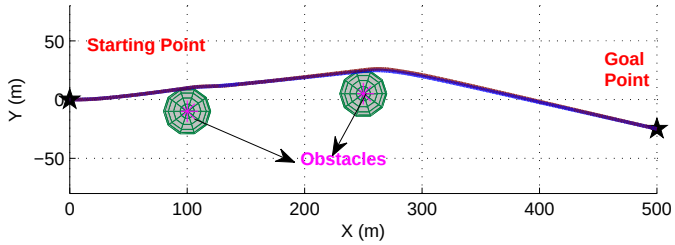


Figure 4.14: 2D view of the trajectory for obstacle avoidance in X-Y plane

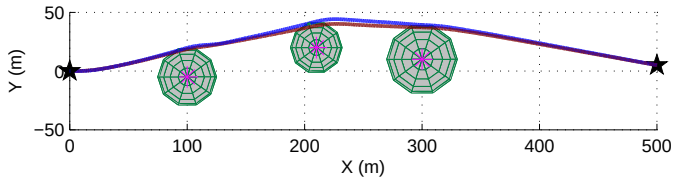


Figure 4.15: 2D view of the trajectory for obstacle avoidance in X-Y plane

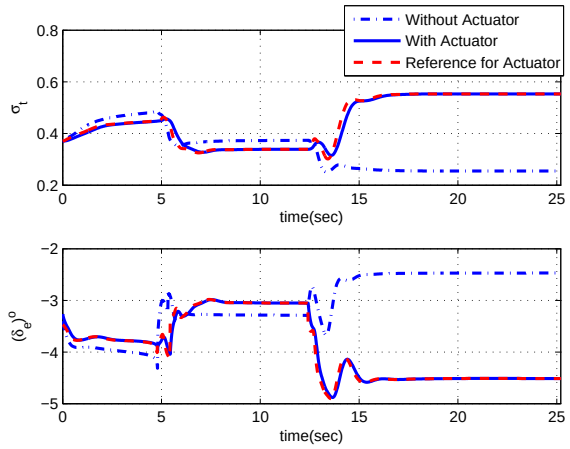


Figure 4.16: Longitudinal control surface deflections of Case 1

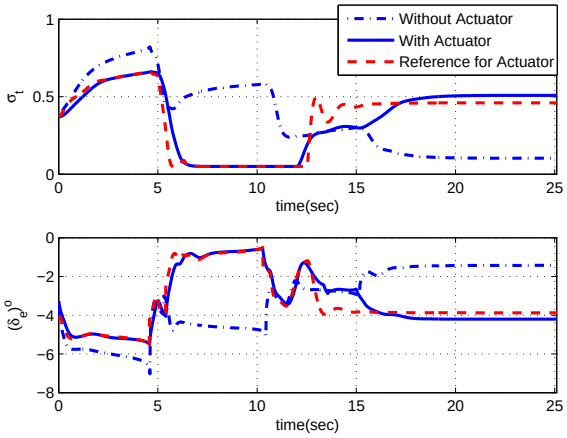


Figure 4.17: Longitudinal control surface deflections of Case 2

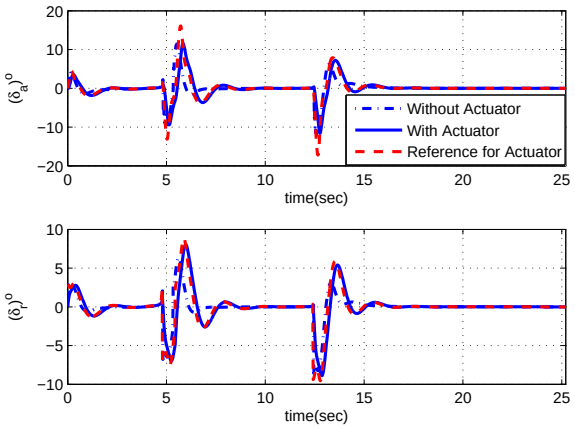


Figure 4.18: Lateral control surface deflections of Case 1

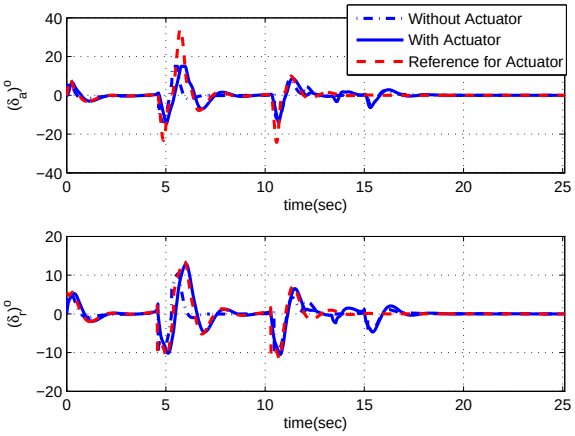


Figure 4.19: Lateral control surface deflections of Case 2

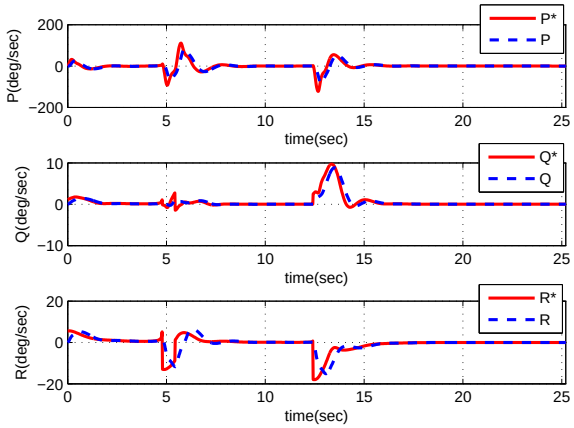


Figure 4.20: Tracking of commanded body angular rates of Case 1

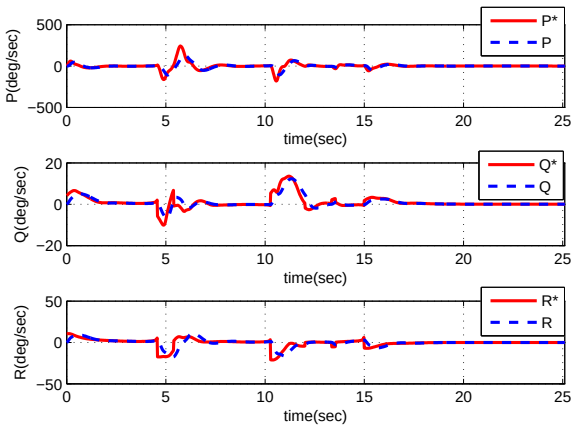


Figure 4.21: Tracking of commanded body angular rates of Case 2

It can be inferred from Fig. 4.20 and Fig. 4.21 that the efficient tracking of commanded body rates is not affected by the number of obstacles present in the scenario even with the actuator in the open loop. Figure 4.22 and Fig. 4.23 shows the profile of total velocity corresponding to the cases, which remain almost constant. It can be depicted from Fig.4.22 and Fig.4.23 that due to the delay introduced by the actuator the deviation of the velocity from its steady state value is higher compared to the case of no actuator in the system. The tracking of the guidance commands in Fig. 4.22 and Fig. 4.23 are executed efficiently, till the goal point is reached even with the open loop actuator model.

Figure 4.24 and Fig. 4.25 shows the tracking of the commanded roll angle for efficient coordinated flight in the presence of the open loop actuator. It can be seen in Fig. 4.24 and Fig. 4.25 that the number of the obstacles do not prevent the attitude of the UAV to get settled to their steady state values even with the actuator in the loop.

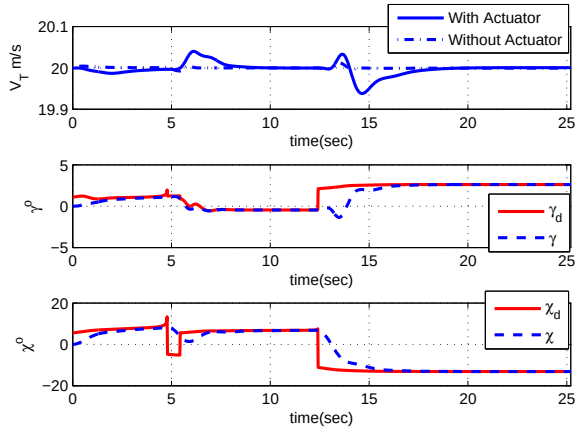


Figure 4.22: Total velocity and its direction of Case 1

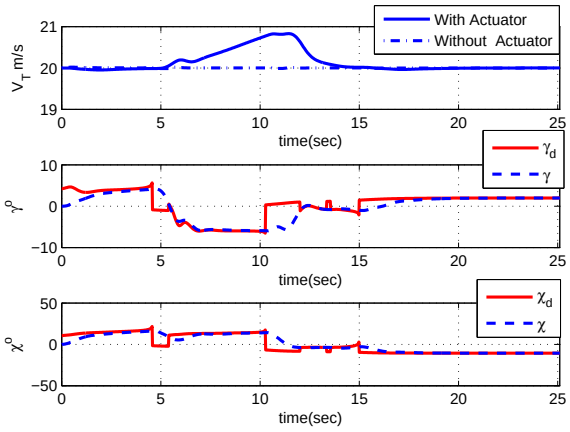


Figure 4.23: Total velocity and its direction of Case 2

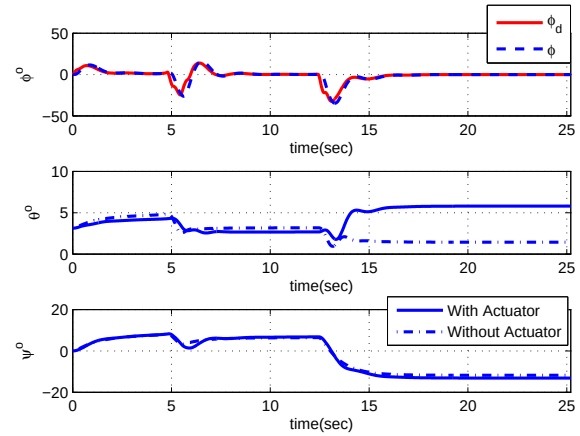


Figure 4.24: Tracking of roll angle and euler angles of Case 1

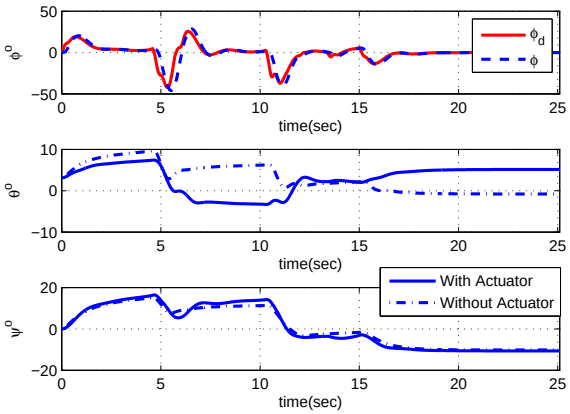


Figure 4.25: Tracking of roll angle and euler angles of Case 2

The above scenarios for single and multiple obstacles were executed with the actuator effect in open loop mode and the effect of actuator in closed loop is not considered. The closed loop mode of the actuator model is considered for the simulation study where the two success conditions for the PIGC algorithm are considered. Simulation studies for various cases under different criteria are tabulated as follows. One of the criteria is perturbed initial state vector of the UAV. The initial condition of the states are randomly perturbed with maximum of $\pm 25\%$. The other criteria is where two constraints are considered, one is the displacement error $< 0.5m$ of the UAV in reaching the goal point and the other is the safety ball incursion $> -1m$ by the UAV. The limit of incursion is equal to half of the wing span of the vehicle. To calculate the appropriate bounds for UAV initial state perturbation, uniform perturbation was given instead of random perturbation. It was found that the maximum bound which the algorithm can tolerate is $+80\%$ and the minimum bound is -50%.

Table 4.3: Different obstacles position and safety ball size with actuator in closed loop mode

Cases	First Obstacle			Second Obstacle			Error at
	Position in m	Ball Size in m	Incursion ($> -1m$)	Position in m	Ball Size in m	Incursion ($> -1m$)	Goal Point ($< 0.5m$)
1	96.8734 4.7687 52.6950	12.0283	0.0155	208.0281 −14.7886 50.7037	6.7051	0.0137	0.0016
2	122.7567 3.3679 58.5233	6.5765	0.7006	227.0221 −4.5857 48.0681	11.6786	0.0001	0.1031
3	85.4898 5.9928 52.1676	8.8094	0.0184	180.8329 −7.1650 48.9617	13.0583	0.0200	0.1584
4	113.6618 4.7648 57.6462	5.9183	2.8161	170.4597 −6.8207 51.0911	16.2983	0.0052	0.1359
5	75.3207 −1.2372 55.4756	10.4306	0.0689	184.4136 −13.9456 49.0004	7.7968	0.0071	0.1988

Table 4.3 represents the case with the position of the obstacle as well as size of the safety ball is varying over all the cases. The initial condition of UAV is equal to the trim conditions. It can be seen from the Table 4.3 that even with the actuator, the UAV reaches well within

the tolerance for the goal point without violating the safety ball beyond the specified limit (equal to half the length of the wing span). Table 4.4 represents the case in which initial condition of the UAV is perturbed from the trim condition. Moreover, the position of the obstacle as well as size of the safety ball is varying over all the cases. It can be seen from the Table 4.4 that with all different variation and with the actuator, the UAV reaches within the tolerance for the goal point without violating the safety ball beyond the specified limit. It can be seen from Tables 4.3, 4.4 that in all the different cases considered, the incursion limit is not violated and the goal point is achieved successfully.

Table 4.4: Different cases with actuator in closed loop mode

Initial Condition of UAV	First Obstacle			Second Obstacle			Error at Goal Point ($< 0.5m$)
	Position in m	Ball Size in m	Incursion ($> -1m$)	Position in m	Ball Size in m	Incursion ($> -1m$)	
+23.24%	96.8734 4.7687 52.6950	12.0283	1.7445	208.0281 −14.7886 50.7037	6.7051	0.8936	0.1428
+15.01%	122.7567 3.3679 58.5233	6.5765	0.0054	227.0221 −4.5857 48.0681	11.6786	2.7044	0.0344
+22.94%	85.4898 5.9928 52.1676	8.8094	4.9406	180.8329 −7.1650 48.9617	13.0583	0.0291	0.0048
+8.94%	113.6618 4.7648 57.6462	5.9183	0.0438	170.4597 −6.8207 51.0911	16.2983	0.0425	0.0449
-11.19%	75.3207 −1.2372 55.4756	10.4306	1.0127	184.4136 −13.9456 49.0004	7.7968	0.0196	0.1421

Simulation studies for larger number of cases have been executed with various combinations exhibited in the above tables and are represented pictorially for the better insight. A case study has been conducted, where the obstacles position and safety ball size around the obstacle are altered. The simulation study is carried out for 200 simulations with two obstacles in the scenario. The same study was carried out for larger simulations and with more number of obstacles in the environment which is not shown for the clarity of results.

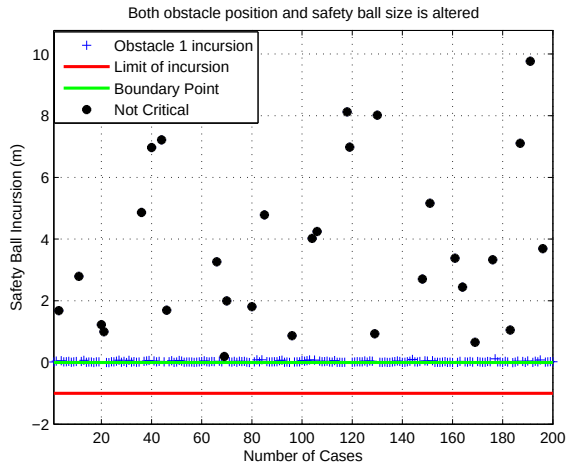


Figure 4.26: Safety Ball Incursion by First Obstacle

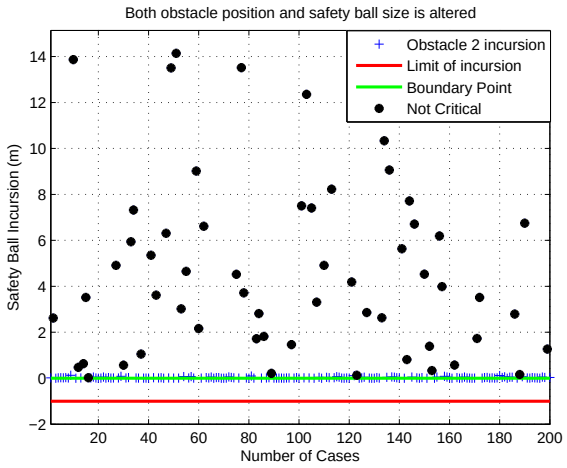


Figure 4.27: Safety Ball Incursion by Second Obstacle

- Case Study: Both obstacle position and safety ball size is altered In this case, UAV initial conditions are same as trim conditions. Here, also a safety margin of $1m$ around the safety ball is considered as the criteria for the case to be a success. Figure 4.26 shows that for all the cases, the first obstacle is not crossing the limit of incursion. Figure 4.27 shows that for all the cases, the second obstacle is also quite away from the limit of the incursion. The second criteria for the case to be a success is that the goal point should be reached within the tolerance bound of $\pm 0.5m$. Figure 4.28 shows that the error in X-direction in reaching the goal point is well within the bound. Similarly, the error in Y and Z-direction in reaching the goal point is also well within the bound as shown in Fig. 4.29 and Fig. 4.30. The success rate of 200 simulations was 100% in this case study.

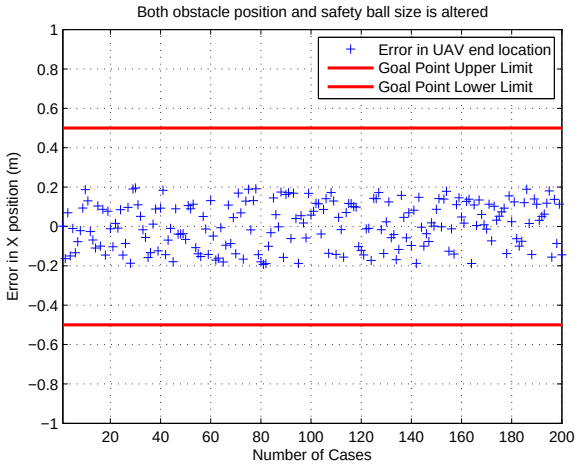


Figure 4.28: Error in reaching the Goal Point in X-direction

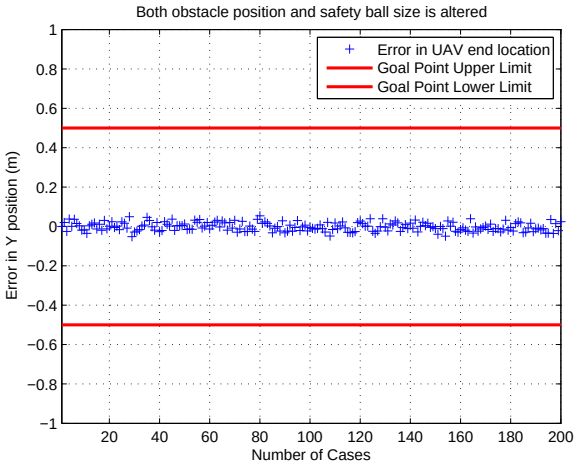


Figure 4.29: Error in reaching the Goal Point in Y-direction

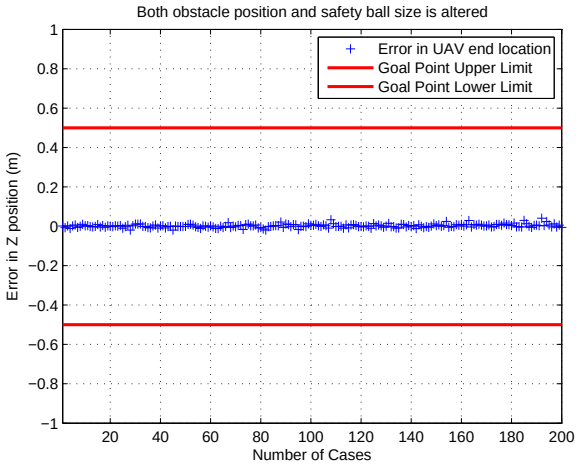


Figure 4.30: Error in reaching the Goal Point in Z-direction

Chapter 5

Neuro-Adaptive Design for PIGC

A control law like NDI in case of PIGC design should perform well for a given flight envelope and in the presence of failure conditions and uncertainty. The major source of uncertainty in modeling rigid, aircraft dynamics is due to lack of knowledge regarding the effects of the nonlinear, unsteady aerodynamics. Albeit, the NDI technique has evolved as a promising tool for nonlinear control design substituting the extensive gain scheduling approach, there are a few critical issues with respect to the technique as well. One of the common issues are modeling errors and parameter inaccuracies due to unsteady aerodynamics. It leads to partial cancelation of the nonlinearities due to inversion of the model which makes the technique sensitive to the parameter uncertainties and hence, there is a need to augment this technique with adaptive/robust control design tools [27]. Aerodynamic and inertia parameter inaccuracies are addressed by reinforcing the NDI technique with a neuro-adaptive design approach [28].

In this section, we present a neuro-adaptive control design, which is capable of addressing the issue of parameter inaccuracy in the model. The philosophy of the approach lies with the fact that the difference of parameter values from their nominal values essentially generate unknown algebraic terms in the model. These unknown functions are captured by neural networks, which are trained online [26]. The methodology of neuro-adaptive design is broadly carried out in two steps: (i) synthesis of a set of neural networks which capture matched unmodelled (neglected) dynamics or model uncertainties because of parametric variations and (ii) synthesis of a controller that drives the state of the actual plant to that of a desired nominal model.

5.1 Actual and Nominal Six-DOF model

Under the assumptions of airplane to be a rigid body and earth to be flat the complete set of Six-DOF equations of motion in body axes system are given by the following differential equations [20], [33].

$$\dot{U} = RV - QW - g\sin\theta + X_a + X_t \tag{5.1}$$

$$\dot{V} = PW - RU + g\sin\phi\cos\theta + Y_a \tag{5.2}$$

$$\dot{W} = QU - PV + g\cos\phi\cos\theta + Z_a \tag{5.3}$$

$$\dot{P} = c_1RQ + c_2PQ + c_3L_a + c_4N_a \tag{5.4}$$

$$\dot{Q} = c_5PR + c_6(R^2 - P^2) + c_7(M_a - M_t) \tag{5.5}$$

$$\dot{R} = c_8PQ - c_2RQ + c_4L_a + c_9N_a \tag{5.6}$$

$$\dot{\phi} = P + Q\sin\phi\tan\theta + R\cos\phi\tan\theta \tag{5.7}$$

$$\dot{\theta} = Q\cos\phi - R\sin\phi \tag{5.8}$$

$$\dot{\psi} = Q\sin\phi\sec\theta + R\cos\phi\sec\theta \tag{5.9}$$

$$\dot{x_i} = U\cos\theta\cos\psi + V(\sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi) + W(\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi) \tag{5.10}$$

$$\dot{y_i} = U\cos\theta\sin\psi + V(\sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi) + W(\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi) \tag{5.11}$$

$$\dot{h_i} = U\sin\theta - V\sin\phi\cos\theta - W\cos\phi\cos\theta \tag{5.12}$$

Six-DOF represented by Eq.(5.1)- Eq.(5.12) forms the nominal plant with nominal aerodynamic coefficients in the force and moments in the body axes of the vehicle. Perturbed plants are formed by giving different percentage of perturbation to the nominal parameters which are more susceptible to uncertainties. The parameters comprises of moment of inertia terms, aerodynamic force coefficients and moment coefficients terms. Six-DOF model consists of six aerodynamic coefficients C_X,C_Y,C_Z,C_l,C_m,C_n which are obtained by the polynomial fit of many aerodynamic derivatives [33].

To design an actual model, uncertainty is added directly to the aerodynamic derivatives present in the aerodynamic coefficients. The total number of aerodynamic derivatives are 85 which leads to 2^{85} combinations. So to achieve a particular set of perturbation to the set of parameters, number of combinations are also randomized. The randomness in combination

were biased such that the parameters were perturbed randomly under uniform distribution and very often they were altered by the corner values(bounds) for the perturbation.

5.2 Weight Training Through Flight Maneuvers

Some pre-knowledge of the behavior of the plant dynamics will help to decide the range of the weights appropriate for the network training. Pre-knowledge is gained in terms of pre-trained stable weights. Test maneuvers like lateral maneuver, longitudinal maneuver, and combined lateral and longitudinal maneuver of UAV were conducted to achieve stable weights. Stabilized weights obtained from preflight maneuvers leads to faster convergence and avoid any misapprehensions in case of actual obstacle avoidance scenario.

5.2.1 Lateral Maneuver

In the lateral maneuver, the objective is to execute coordinated turns at constant height. The lateral turns demands change in heading angle with zero climb rate. The guidance command becomes the desired heading angle along with the desired pitch angle and desired roll angle which allows UAV to execute coordinated turns at constant height. The desired heading angle is observed in terms of step command or a sinusoidal command. The desired pitch angle is generated by enforcing first order error dynamics associated with climb rate($\dot{h}$) to go to zero through NDI. Similarly, the desired roll angle is generated by enforcing the first order error dynamics associated with side velocity (V) to go to zero which ensures a coordinated turn. The step command follows a patterns as $\dot{\chi}_d = 0.8 deg/sec$ for the time interval $(0-12)$sec, $-0.4 deg/sec$ for the interval $(12-20)$sec, $0.2 deg/sec$ for the interval $(20-24)$sec, $-0.2 deg/sec$ for the interval $(24-28)$ sec and $0 deg/sec$ for $t > 28$ sec. The turn rate remains constant within a particular interval and heading angle becomes the ramp command. In case of a sinusoidal command, the turn rate is changing sinusoidally and so the heading angle changes co-sinusoidally which can be stated as

$$\begin{aligned} \dot{\chi}_d &= \left(\frac{\sin(2\pi t_k)}{T}\right) + \zeta \\ \chi_d &= \left(\frac{T}{2\pi}\right)\left[\cos\left(\frac{2\pi t_{k-1}}{T}\right) - \cos\left(\frac{2\pi t_k}{T}\right)\right] \end{aligned}$$

where ζ is the phase angle of the signal $\dot{\chi}_d$ and T is the time period of the commanded signal.

5.2.2 Longitudinal Maneuver

In case of a longitudinal maneuver, the objective is to execute a climb at a constant climb rate with no heading corrections. The desired flight path angle is observed in terms of step command and sinusoidal command. The step command follows a patterns as $\gamma_d = 0.8deg$ for the time interval $(0-12)$sec, $-0.4deg$ for the interval $(12-20)$sec, $0.2deg$ for the interval $(20-24)$sec, $-0.2deg$ for the interval $(24-28)$ sec and $0deg$ for $t > 28$ sec. The sinusoidal command is given by

$$\begin{aligned} \dot{\gamma}_d &= \left(\frac{\sin(2\pi t_k)}{T}\right) + \xi \\ \gamma_d &= \left(\frac{T}{2\pi}\right)\left[\cos\left(\frac{2\pi t_{k-1}}{T}\right) - \cos\left(\frac{2\pi t_k}{T}\right)\right] \end{aligned}$$

where ξ is the phase angle of the signal $\dot{\gamma}_d$. In this exercise, the desired heading angle and the desired roll angle is set to zero.

5.2.3 Combined Lateral and Longitudinal Maneuver

In case of a combined maneuver, the objective is to execute turns both in the horizontal and vertical planes respectively. These turns demand change in the heading angle $\dot{\chi}_d$ and and change in the flight path angle $\dot{\gamma}_d$. The guidance commands are the desired heading angle, the desired climb rate and the desired roll angle which allows UAV to execute coordinated turns at varying height. The desired heading angle and the desired climb rate are observed in terms of sinusoidal commands. The desired roll angle is generated by enforcing the first order error dynamics associated with the side velocity (V) to go to zero through NDI which ensures a coordinated turn. In case of sinusoidal commands, the turn rate and the climb rate are changing sinusoidally and so the heading angle and the flight path angle changes co-sinusoidally which are stated as follows

$$\begin{aligned} \dot{\chi}_d &= \left(\frac{\sin(2\pi t_k)}{T_1}\right) + \zeta \\ \chi_d &= \left(\frac{T_1}{2\pi}\right)\left[\cos\left(\frac{2\pi t_{k-1}}{T_1}\right) - \cos\left(\frac{2\pi t_k}{T_1}\right)\right] \\ \dot{\gamma}_d &= W_1 + \left(\frac{\sin(2\pi t_k)}{T_2}\right) + \xi \\ \gamma_d &= W_1\, t_k + \left(\frac{T_2}{2\pi}\right)\left[\cos\left(\frac{2\pi t_{k-1}}{T_2}\right) - \cos\left(\frac{2\pi t_k}{T_2}\right)\right] \end{aligned}$$

where T_1 and T_2 are time periods of the commanded signals in horizontal and vertical plane respectively. W_1 is the bias value which is added to keep the commanded climb rate always positive.

5.2.4 Neuro-Adaptive Design for Flight Maneuvers

For the present study, the actual plant model is generated by considering $\pm 20\%$ random perturbation both in the moment of inertia terms and aerodynamics coefficients about their nominal values. Neuro-adaptive (NA) controller is designed for the inner loop of the actual plant model to overcome the uncertainty in the aerodynamic coefficients. The inner loop comprises of the body angular rates which are functions of aerodynamic coefficients and the control surface deflections appearing in the affine form as given in Eq.(5.4)- Eq.(5.6). As the control is appearing in body rate dynamics, it will be directly affected by the aerodynamic uncertainties in the actual plant. The robustness is added to the inner loop where the states and the output are roll, pitch and yaw body rates. It is ensured that by applying the neuro-adaptive controller only to the inner loop, the nonlinear and distributed uncertainties of the aerodynamic coefficients in complete Six-DOF model is taken into account. The objective of the inner loop as stated in nominal PIGC design was $[P_d \quad Q_d \quad R_d] \rightarrow [P^* \quad Q^* \quad R^*]$ as $t \rightarrow \infty$, where $[P_d \quad Q_d \quad R_d]$ represents the nominal states without any uncertainty in the plant model and $[P^* \quad Q^* \quad R^*]$ are the commanded values generated from the outer guidance loop. In case of pre-flight test maneuvers, the guidance commands are open loop commands which are not evaluated based on the feedback from the actual states of the perturbed plant. Hence, the inner loop robustness is independent of the guidance commands. The objective of the NA controller is to make the states of the actual plant $[P \quad Q \quad R] \rightarrow [P_d \quad Q_d \quad R_d] \rightarrow [P^* \quad Q^* \quad R^*]$. Considering the output dynamics of the inner loop of the nominal plant, the system is of the form $\dot{Y}_d = f_{Y_d}(X_d) + G_{Y_d}(X_d)U_d$, where $\dot{Y}_d = [\dot{P} \quad \dot{Q} \quad \dot{R}]$. Using NDI, the inner loop can be resolved for the nominal controller as

$$U_d = [G_{Y_d}(X_d)]^{-1}[f_{Y_d}(X_d) + K_d(Y_d - Y^*)] \tag{5.13}$$

where K_d is the positive definite gain vector and selected from the settling time (T_s) of the system dynamics.

In case of actual plant, the output dynamics differs from the nominal output dynamics

for the inner loop because the dynamics contains nominal parameters along with their corresponding uncertainties and inaccuracies. The actual output dynamic equation will then be given by

$$\dot{Y} = f_Y(X) + G_Y(X)U + d_Y(X) \tag{5.14}$$

where $d_Y(X)$ is an unknown function comprises of uncertainties of the perturbed coefficients and hence needs to be captured by designing a neuro-adaptive controller. The unknown function $d_Y(X)$ is captured through neural network function approximation. The design for the approximation of the unknown function results in an approximate output dynamics with function approximation as $\hat{d_Y}(X)$ which is given by

$$\dot{Y_a} = f_{Y_d}(X) + G_{Y_d}(X)U + \hat{d_Y}(X) + K_a(Y - Y_a) \tag{5.15}$$

By substituting, $\dot{Y_a}$ dynamics at the components level, it can be given as

$$\begin{aligned}\begin{bmatrix} \dot{P}_a \\ \dot{Q}_a \\ \dot{R}_a \end{bmatrix} &= \begin{bmatrix} c_1RQ + c_2PQ + c_3L_{a_x} + c_4N_{a_x} \\ c_5PR + c_6(P^2 - R^2) + c_7(M_{a_x} - M_t) \\ c_8PQ - c_2RQ + c_4L_{a_x} + c_9N_{a_x} \end{bmatrix} + \begin{bmatrix} c_3L_{a_u} & 0 & c_4N_{a_u} \\ 0 & c_7M_{a_u} & 0 \\ c_4L_{a_u} & 0 & c_9N_{a_u} \end{bmatrix} U \\ &+ \begin{bmatrix} \hat{d}_p(X) \\ \hat{d}_q(X) \\ \hat{d}_r(X) \end{bmatrix} + K_a \begin{bmatrix} P - P_a \\ Q - Q_a \\ R - R_a \end{bmatrix}\end{aligned} \tag{5.16}$$

where,

$$\begin{aligned} L_{a_x} &\triangleq \bar{q}Sb[C_{l_\beta}(\alpha)\ \beta + C_{l_P}(\alpha)\ P + C_{l_R}(\alpha)\ R] \\ M_{a_x} &\triangleq \bar{q}Sc[C_{m_0} + C_{m_\alpha}(\alpha)\ \alpha + C_{m_\beta}(\alpha, \beta)\ \beta + C_{m_Q}(\alpha)\ Q] \\ N_{a_x} &\triangleq \bar{q}Sb[C_{n_\beta}(\alpha)\ \beta + C_{n_P}(\alpha)\ P + C_{n_R}(\alpha)\ R] \end{aligned}$$

and,

$L_{a_u} \triangleq \bar{q}SbC_{l_{\delta a}}$; $M_{a_u} \triangleq \bar{q}ScC_{m_{\delta e}}$; $N_{a_u} \triangleq \bar{q}SbC_{n_{\delta r}}$

Finally, the adaptive controller U is designed, by enforcing first order error dynamics so as to ensure that $Y_a \rightarrow Y_d$. The first order error dynamics is written as

$$(\dot{Y_a} - \dot{Y_d}) + K(Y_a - Y_d) = 0 \tag{5.17}$$

$$\begin{bmatrix} \dot{P}_a \\ \dot{Q}_a \\ \dot{R}_a \end{bmatrix} - \begin{bmatrix} \dot{P}_d \\ \dot{Q}_d \\ \dot{R}_d \end{bmatrix} + K \left(\begin{bmatrix} P_a - P_d \\ Q_a - Q_d \\ R_a - R_d \end{bmatrix} \right) = 0 \tag{5.18}$$

Using Eq. (5.15) and substituting $\dot{Y}_a$ dynamics at the components level from Eq. (5.16) in Eq.(5.18) with necessary algebra, the adaptive control is obtained as given in Eq. (A.31).

$$U = -[G_{Y_d}(X)]^{-1}(f_{Y_d}(X) + \hat{d_Y}(X) + K_a(Y - Y_a) - f_{Y_d}(X_d) - G_{Y_d}(X_d)U_d + K(Y_a - Y_d)) \tag{5.19}$$

The adaptive control $U = [\delta_e \ \delta_a \ \delta_r]$ in Eq.(5.19) along with the throttle control σ_t are used for system propagation of the actual plant.

Next, the neural network selection for function approximation with $\hat{d_Y}(X)$ and its training is discussed. The first step in this regard is to select an appropriate basis function (ϕ) [24], [26]. The selection of the basis function plays a vital role in online training. Note that, the magnitudes of the uncertain parameters in the actual system equations may be of different orders. In such a case, having one network approximation for the uncertainties of the whole system may affect the convergence of the single network. Therefore, an important concept used in this work is to separate all the channels such that there will be n independent neural networks to approximate uncertainties in each of the n channels. It also facilitates easier mathematical analysis of the network consists of the states, the control vector and the slack variable vector.

In the current problem, the inner control loop has states which are taken as output for the formulation. Hence, it is assumed that the basis function is a function of the output (Y, Y_a). So we assume that the unknown function $d_Y(X)$ can be represented in terms of the basis function vector $\phi(Y, Y_a)$ as given in Eq. (A.16).

$$\hat{d_{y_i}}(X) = \hat{W_i}^T \phi_i(Y, Y_a) \tag{5.20}$$

The basis function vector in each channel is chosen as follows

$$\phi_P(X) = \begin{bmatrix} La \\ Na \\ c_3 \ \bar{q} S b \ \beta \ \alpha \\ c_3 \ \bar{q} \ S \ b \ P \ b2v \ \alpha \\ c_3 \ \bar{q} \ S \ b \ R \ b2v \ \alpha \\ c_4 \ \bar{q} \ S \ b \ \beta \end{bmatrix} \tag{5.21}$$

$$\phi_Q(X) = \begin{bmatrix} M_a \\ M_t \\ c_7\ \bar{q}\ S\ c\ \alpha \\ c_7\ \bar{q}\ S\ c\ \beta^2 \\ c_7\ \bar{q}\ S\ c\ Q\ c2v \\ c_7\ \bar{q}\ S\ c\ Q\ c2v\ \alpha \end{bmatrix} \tag{5.22}$$

$$\phi_R(X) = \begin{bmatrix} La \\ Na \\ c_9\ \bar{q}\ S\ b\ \beta\ \alpha \\ c_9\ \bar{q}\ S\ b\ P\ b2v\ \alpha \\ c_9\ \bar{q}\ S\ b\ R\ b2v\ \alpha \\ c_4\ \bar{q}\ S\ b\ \beta \end{bmatrix} \tag{5.23}$$

where, $b2v = (\frac{b}{2V_T})$, $c2v = (\frac{c}{2V_T})$, $L_a = \bar{q}\ S\ b\ C_l$, $M_a = \bar{q}\ S\ c\ C_m$, $N_a = \bar{q}\ S\ b\ C_n$ and $M_t = d\ (T_{max}\ \sigma_t)$ in respective channels. d is the offset of the thrust line from the CG of the vehicle. C_l,C_m,C_n moment coefficients as expressed earlier are given by

$$C_l = C_{l_\beta}(\alpha)\beta + C_{l_{\delta_a}}(\alpha)\delta_a + C_{l_P}(\alpha)\bar{P} + C_{l_R}(\alpha)\bar{R} \tag{5.24}$$

$$C_m = C_{m_0} + C_{m_\alpha}(\alpha)\alpha + C_{m_\beta}(\alpha,\beta)\beta + C_{m_{\delta_e}}(\alpha)\delta_e + C_{m_Q}(\alpha)\bar{Q} \tag{5.25}$$

$$C_n = C_{n_\beta}(\alpha)\beta + C_{n_{\delta_r}}(\alpha)\delta_r + C_{n_P}(\alpha)\bar{P} + C_{n_R}(\alpha)\bar{R} \tag{5.26}$$

where,

$$[\bar{P}\quad \bar{Q}\quad \bar{R}] \;=\; \frac{1}{2V_T}[bP\quad cQ\quad bR]$$

In each of the channel, P, Q, R are the actual states. It can be seen from Eq. (5.24)-Eq.(5.26), that the coefficients depends on the control surface deflections. In the definition of the basis functions, the coefficients contain the previous step value of the adaptive control.

Different test cases were executed which in terms of the step and sinusoidal commands. Finite/pre-trained weights obtained from the various test maneuvers are the stabilized weights used for obstacle avoidance. Stabilized weights are obtained by taking average of the weights getting updated during the last 10 seconds of the flight maneuver. Test maneuvers like step and sinusoidal were performed but for compactness, the results of the lateral maneuver with sinusoidal guidance commands are demonstrated. The tracking of $Y \rightarrow Y_a \rightarrow Y^*$ is shown directly in the plots instead of $Y \rightarrow Y_a \rightarrow Y_d \rightarrow Y^*$ which is explained in the next section [5.3].

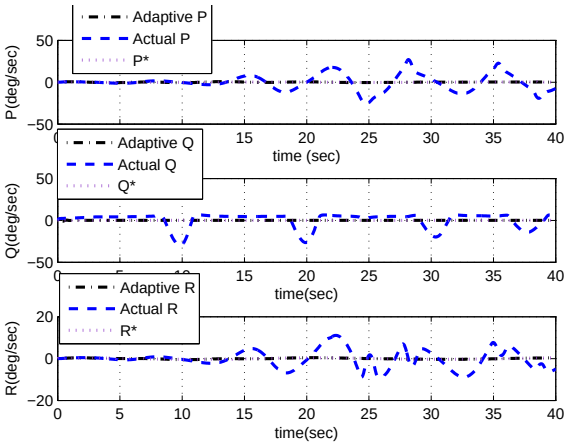


Figure 5.1: Body angular rates tracking

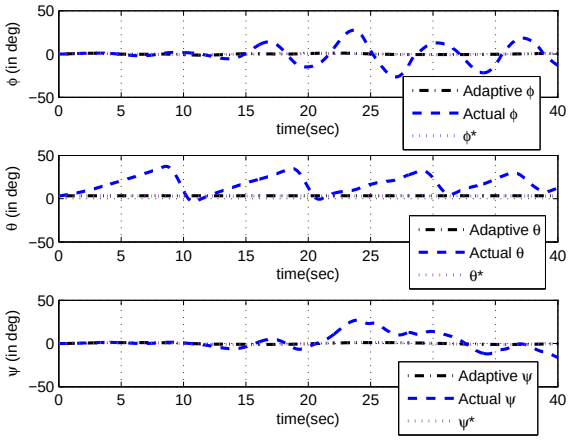


Figure 5.2: Guidance command tracking

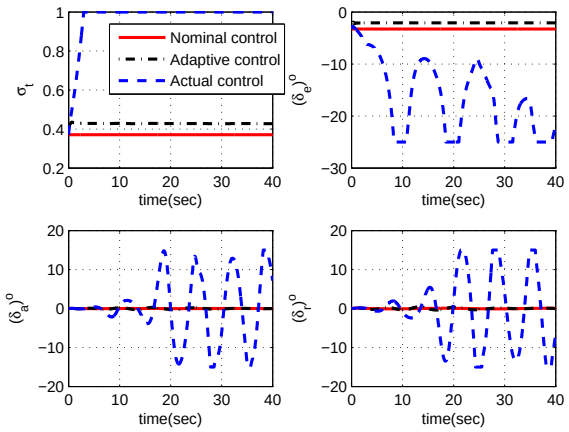


Figure 5.3: Control surface deflections

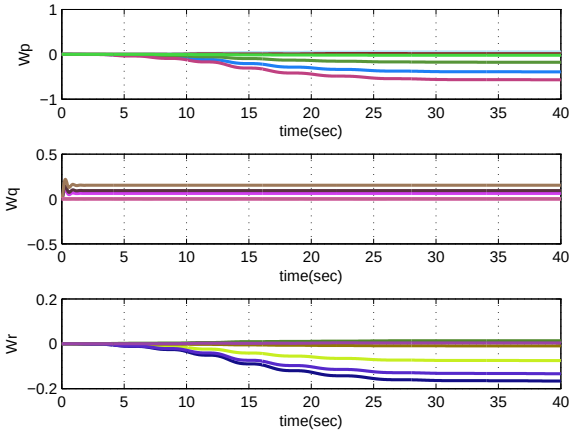


Figure 5.4: Weights of each output channel

It can be seen from Fig. 5.1 that the actual body rates are not able to track the commanded body rates whereas the adaptive body rates are able to track the commanded values. Even in Fig. 5.2 the attitude angles generated by the actual model are not able to follow the guidance commands. Figure 5.3 shows the control surface deflections, in which the throttle saturates in case of nominal control when applied to actual model. The elevator deflection switches in between the saturation compared to the adaptive elevator control which tries to follow the nominal control of the nominal plant. Similarly, the aileron and the rudder are set into oscillations when nominal control is applied to the actual model compared to adaptive control which is almost similar to the nominal control of the nominal plant. It can seen from the Fig. 5.4 that the weights corresponding to each of the output channels $[P\ Q\ R]$ are stabilized very soon to a steady state value.

5.3 Obstacle Avoidance using Neuro-Adaptive Augmented PIGC Design

In case of PIGC design, as we discussed earlier in chapter [4], it consists of two loops, the outer loop executes the guidance commands tracking and the inner loop executes the angular body rates tracking. PIGC is augmented with neuro-adaptive design so as to make it robust against the parameter inaccuracies of the plant. It can be seen from Fig. 5.5 that the guidance commands are calculated based on the feedback of the states of the nominal plant, unlike the test maneuvers, where the guidance commands are given as open loop commands. The existing model for NA design as shown in Fig. 5.5 is much suited to the problems where the guidance commands is independent of the vehicle state information. Note that, in case of the obstacle avoidance problem, the path of the vehicle will get perturbed due to the plant uncertainty. Since, the guidance commands depend on the relative position between the vehicle and the obstacle so they should be calculated based on the actual states of the perturbed plant. Therefore, the guidance commands will be affected by the inner loop robustness. It implies that the algorithm will work in closed loop with guidance update based on the feedback of the actual states as shown in Fig. 5.6.

Now, the problem definition is modified as the generation of guidance command does not depend on the nominal states but on the actual states as shown in Fig. 5.6. The modification of closed loop NA design can be stated as

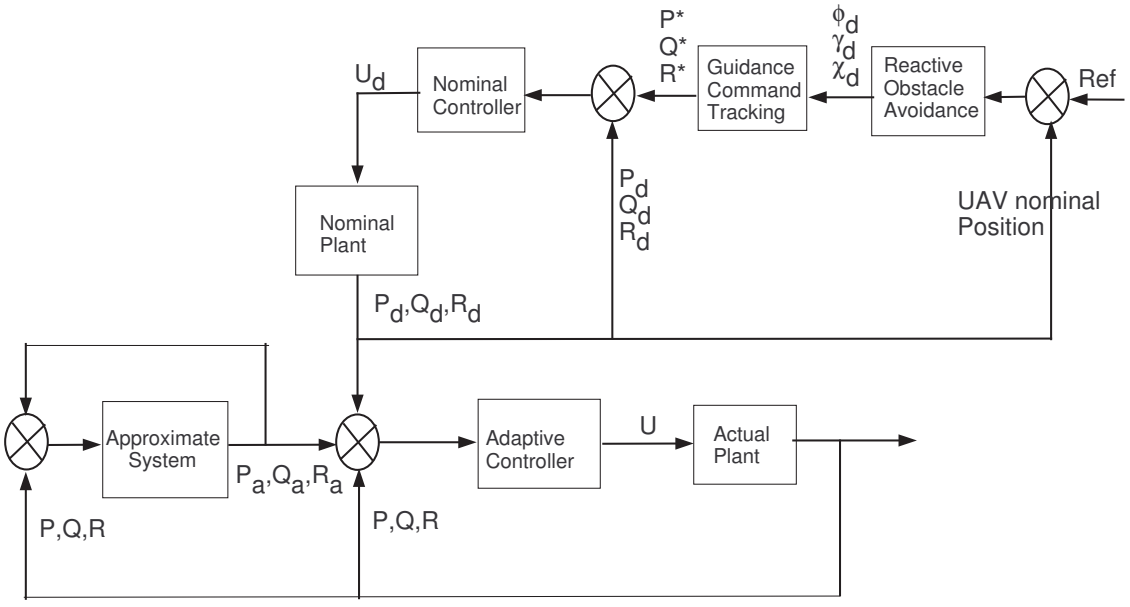


Figure 5.5: Open loop neuro-adaptive design

- The relative position of the vehicle from the obstacle will always account for the actual position of the vehicle.
- The actual position of the vehicle gives rise to the guidance commands $[\phi_{dac},\ \gamma_{dac},\ \chi_{dac}]$ unlike the guidance commands$[\phi_d,\ \gamma_d,\ \chi_d]$ of the nominal plant.
- The guidance commands $[\phi_{dac},\ \gamma_{dac},\ \chi_{dac}]$ generates the commanded body rates $Y^*_{ac} \neq Y^*$ (corresponds to the guidance commands$[\phi_d,\ \gamma_d,\ \chi_d]$).
- This implies Y_d loses its meaning in case of closed loop NA design as the definition of commanded body rates Y^*_{ac} is changing at every time step unlike Y^* which was fixed in case of open loop NA design as shown in Fig. 5.5

Therefore, the definition of the nominal system is altered due to the closed loop action in case of the NA design. The closed loop action leads to a pseudo-nominal system in which control surface deflections for the nominal parameters of the system are generated based on Y^*_{ac}. Now, the objective of closed loop NA controller for the problem of obstacle avoidance is renewed as $Y \rightarrow Y_a \rightarrow Y^*_{ac}$ as shown in Fig. 5.6. It ensures fast tracking of the output

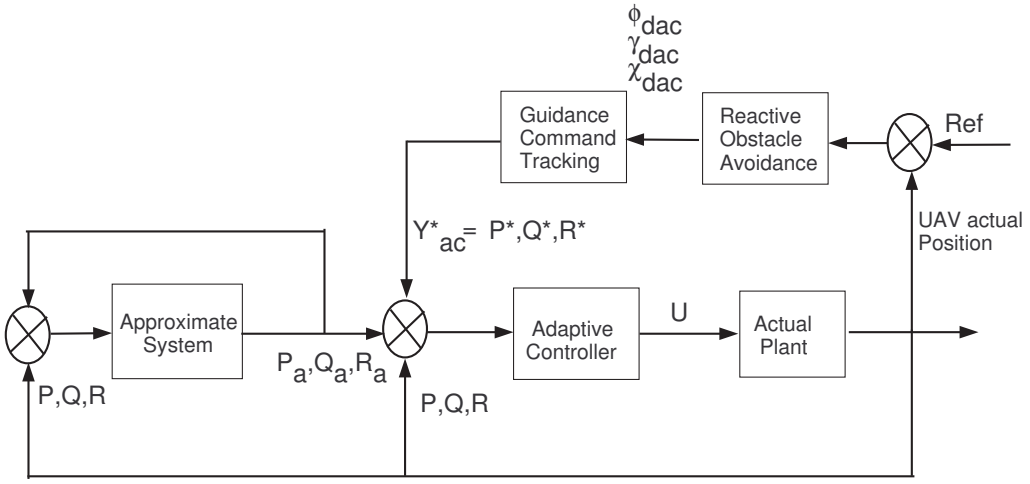


Figure 5.6: Closed loop neuro-adaptive design

$Y \rightarrow Y^*_{ac}$ with minimum delay. The first order error dynamics equation now becomes

$$(\dot{Y}_a - \dot{Y}^*_{ac}) + K_g(Y_a - Y^*_{ac}) = 0 \tag{5.27}$$

With the assumption, $\dot{Y}^*_{ac} = 0$, Eq.(5.32) at the component level becomes

$$\begin{bmatrix} \dot{P}_a \\ \dot{Q}_a \\ \dot{R}_a \end{bmatrix} + K_g \left(\begin{bmatrix} P_a - P^* \\ Q_a - Q^* \\ R_a - R^* \end{bmatrix} \right) = 0 \tag{5.28}$$

By substituting $\dot{Y}_a$ dynamics from Eq.(5.16), the adaptive controller U can be derived as

$$U = -[G_{Y_d}(X)]^{-1}(f_{Y_d}(X) + \hat{d_Y}(X) + K_a(Y - Y_a) + K_g(Y_a - Y^*_{ac})) \tag{5.29}$$

The adaptive control $U = [\delta_e \ \delta_a \ \delta_r]$ along with the throttle control σ_t are used for the actual system propagation.

The first step of the NA design involves function approximation on the principles of the neural network with enforcement of $Y \rightarrow Y_a$. In case of obstacle avoidance, $\hat{d_Y}(X)$ calculation is initiated with finite weights instead of zero weights and with same basis functions as given by Eqs. (5.21),(5.22) and (5.23)for the respective output channels. The finite weights are the pre-trained weights obtained from the pre-flight maneuvers which were executed with zero weights initialization. The concept of pre-trained weights as initial weights helps in faster

online training of the network. Moreover, it also acquires the knowledge about the plant uncertainties with off-line training of weights using different pre-flight maneuvers.

Note that, the guidance commands as shown in Fig. 5.6 consists of ϕ_{dac}, γ_{dac}, χ_{dac} in which γ_{dac}, and χ_{dac} are calculated geometrically from the relative position of the UAV and the obstacle. However, the ϕ_{dac} command for the coordinated turn is calculated by the enforcement of the first order error dynamics of the side velocity V. The $\dot{V}$ dynamics contains force term which is susceptible to the aerodynamic uncertainty, hence there is a need to define an adaptive ϕ_{dac} command. The adaptive ϕ_{dac} command is derived based on the closed loop NA design as discussed in the following section. The basis function vector for the $\dot{V}$ dynamics is given by

$$\phi_V(X) = \begin{bmatrix} \frac{\bar{q}\,S}{m}\ \beta\ \alpha \\ \frac{\bar{q}\,S}{m}\ P\ b2v\ \alpha \\ \frac{\bar{q}\,S}{m}\ R\ b2v\ \alpha \\ c_9\ \bar{q}\ S\ b\ \beta\ \alpha \\ \frac{\bar{q}\,S}{m}\ \delta_r \\ \frac{\bar{q}\,S}{m}\ \alpha\ \delta_a \end{bmatrix} \tag{5.30}$$

In case of adaptive ϕ_{dac} command, the zero weight initialization is adopted for function approximation $\hat{d}_v(X)$. The $\dot{V}_a$ dynamics approximating the actual $\dot{V}$ dynamics can be written as

$$\dot{V}_a \;=\; f_{v_d}(X) + g_{v_d}(X)\phi_{dac} + \hat{d}_v(X) + k_{va}(V - V_a) \tag{5.31}$$

The adaptive ϕ_{dac} command is calculated by enforcing first order dynamics such that $V_a \rightarrow V^*$

$$(\dot{V}_a - \dot{V}^*) + k_{vd}(V_a - V^*) = 0 \tag{5.32}$$

With the assumption, $\dot{V}^* = 0$, Eq. (5.32) will become

$$\dot{V}_a \;=\; -k_{vd}(V_a - V^*) \tag{5.33}$$

By substituting, Eq. (5.31) in Eq. (5.33) and with algebraic manipulation we get

$$\phi_{dac} \;=\; -(g_{v_d}(X))^{-1}(k_{vd}(V_a - V^*) + f_{v_d}(X) + \hat{d}_v(X) + k_{va}(V - V_a)) \tag{5.34}$$

The adaptive control U derived in Eq. (5.29) has full bandwidth which is practically unavailable. Therefore, it is passed through the first order actuator model before giving to the actual system. To reduce the effect of the delay on the performance of the system, an actuator controller is designed which treats the adaptive control U as states (assuming U is available for feedback)and generates the available adaptive control of the system after observing the hard constraints like the rate and the position limit.

5.4 Simulation Study

Various simulations have been executed with multiple obstacles of varying size of safety ball around them. The adaptive control is designed and is passed through the closed loop actuator. Comparative results with zero weights and finite weights are demonstrated to show the usefulness of the pre-flight maneuvers. The success of the NA augmented PIGC depends on the efficient and accurate tracking of the commanded angular body rates of the inner loop. Due to the closed loop NA design, the output tracking of the inner loop finally ensures the obstacle avoidance with the tolerance bounds on the safety ball incursion and the goal point attainment.

5.4.1 Control Design Parameters

The gain selection plays an important role while using NDI in cascaded form. In the cascaded form of NDI, different loops have different settling time based on the dynamics involved. In case of nominal PIGC, there are two loops, namely the outer and the inner loop. With the notion of the inner loop dynamics being faster than the outer loop dynamics the following gains for the different test maneuvers have been selected. The gains for the nominal system in case of the lateral maneuver are $k_\phi = 5,\ k_\theta = 8,\ k_\chi = 1,\ k_P = 12,\ k_Q = 15,\ k_R = 5$ and in case of the combined (including lateral and longitudinal) maneuver, gains used are $k_\phi = 5,\ k_\theta = 5,\ k_\chi = 5,\ k_P = 15,\ k_Q = 10,\ k_R = 15$. Finally, for the obstacle avoidance with nominal PIGC, two sets of gains K_1 and K_2 are used in the NA design. $K_1 = (k_V = 5,\ k_U = 1,\ k_\phi = 7,\ k_\gamma = 1,\ k_\chi = 1,\ k_P = 14,\ k_Q = 7,\ k_R = 7)$ and $K_2 = (k_V = 2,\ k_U = 2,\ k_\phi = 3,\ k_\gamma = 1,\ k_\chi = 1,\ k_P = 10,\ k_Q = 7,\ k_R = 7)$. The gain matrix $K_g = diag(12, 10, 10)$ for $Y_a \rightarrow Y^*$ and $K_a = diag(12, 9, 10)$ for $Y \rightarrow Y_a$. The weight update rule has tuning parameters which are different for different maneuvers. In case of

obstacle avoidance, learning rates in all the channels are $gv_{learn} = 10,\ gp_{learn} = 30,\ gq_{learn} = 20,\ gr_{learn} = 100$. The damping coefficients $dv_\sigma = dp_\sigma = dq_\sigma = dr_\sigma = 1e-6$ and the positive coefficients are $p = [p_v = 0.6,\ p_p = 2,\ p_q = 0.7,\ p_r = 0.8]$.

5.4.2 Numerical Results

The actual plant is obtained by giving ±20% random perturbation to the aerodynamic coefficients and inertia terms of the nominal plant. The percentage of perturbation given to all the aerodynamic derivatives is shown in Fig. 5.7.

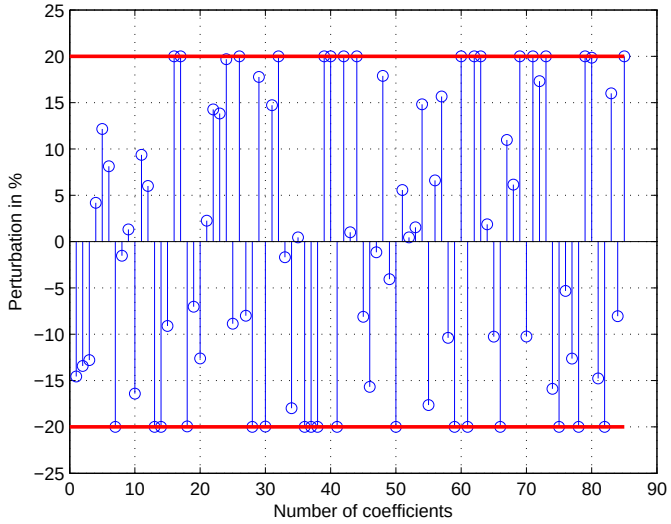


Figure 5.7: Percentage of perturbation for aerodynamic derivatives

Table 5.1: Parameters value and their perturbation

Parameter	Nominal Value	Perturbed Value	Δ Value	Δ in (%)
C_{Z_0}	1.6530e-001	1.8936e-001	-2.4064e-002	-1.4558e+001
z_{10}	8.7138e-002	9.8834e-002	-1.1696e-002	-1.3422e+001
C_{Z_β}	-2.0001e-003	-2.2558e-003	2.5568e-004	-1.2783e+001
$C_{Z_{\delta e}}$	3.9823e-003	3.8154e-003	1.6691e-004	4.1912e+000
z_{11}	-9.1867e-003	-8.0702e-003	-1.1165e-003	1.2153e+001

Continued on next page

Table 5.1 – continued from previous page

Parameter	Nominal Value	Perturbed Value	Δ Value	Δ in (%)
z_{12}	2.4242e-004	2.2273e-004	1.9691e-005	8.1226e+000
z_{20}	6.9303e+000	8.3164e+000	-1.3861e+000	-2.0000e+001
z_{21}	-4.7657e-002	-4.8387e-002	7.2981e-004	-1.5314e+000
C_{X_0}	3.8600e-002	3.8091e-002	5.0932e-004	1.3195e+000
x_{10}	-4.0376e-003	-4.6998e-003	6.6223e-004	-1.6402e+001
x_{11}	-1.0525e-003	-9.5387e-004	-9.8633e-005	9.3713e+000
x_{20}	-3.5832e-004	-3.3679e-004	-2.1526e-005	6.0074e+000
x_{21}	-2.2061e-005	-2.6473e-005	4.4122e-006	-2.0000e+001
x_{30}	-1.8476e-001	-2.2171e-001	3.6952e-002	-2.0000e+001
x_{31}	-1.0227e-001	-1.1158e-001	9.3115e-003	-9.1048e+000
x_{12}	2.7887e-003	2.2310e-003	5.5774e-004	2.0000e+001
x_{13}	1.0917e-004	8.7336e-005	2.1834e-005	2.0000e+001
x_{14}	-5.3586e-006	-6.4272e-006	1.0686e-006	-1.9941e+001
x_{22}	-5.7342e-006	-6.1370e-006	4.0280e-007	-7.0244e+000
C_{m_0}	3.4600e-002	3.8960e-002	-4.3601e-003	-1.2601e+001
m_{10}	-1.3841e-002	-1.3529e-002	-3.1212e-004	2.2550e+000
m_{11}	-2.6206e-004	-2.2468e-004	-3.7380e-005	1.4264e+001
m_{12}	-1.7853e-005	-1.5383e-005	-2.4699e-006	1.3835e+001
m_{13}	-2.1109e-006	-1.6954e-006	-4.1555e-007	1.9686e+001
m_{14}	1.1346e-007	1.2352e-007	-1.0056e-008	-8.8633e+000
m_{20}	2.4049e-004	1.9239e-004	4.8098e-005	2.0000e+001
m_{21}	-7.8566e-006	-8.4852e-006	6.2861e-007	-8.0010e+000
m_{23}	-7.8866e-007	-9.4639e-007	1.5773e-007	-2.0000e+001
y_{22}	1.0663e-006	8.7674e-007	1.8956e-007	1.7778e+001
m_{30}	-1.4500e-002	-1.7395e-002	2.8950e-003	-1.9966e+001
m_{31}	9.2552e-006	7.8926e-006	1.3626e-006	1.4723e+001
m_{32}	9.0437e-006	7.2350e-006	1.8087e-006	2.0000e+001
m_{40}	-1.3954e+001	-1.4189e+001	2.3495e-001	-1.6838e+000
m_{41}	1.7379e-003	2.0505e-003	-3.1258e-004	-1.7986e+001

Continued on next page

Table 5.1 – continued from previous page

Parameter	Nominal Value	Perturbed Value	Δ Value	Δ in (%)
m_{42}	1.6743e-003	1.6666e-003	7.7273e-006	4.6152e-001
y_{10}	9.9319e-003	1.1918e-002	-1.9864e-003	-2.0000e+001
y_{11}	2.9462e-004	3.5354e-004	-5.8924e-005	-2.0000e+001
y_{12}	1.7831e-005	2.1397e-005	-3.5662e-006	-2.0000e+001
y_{13}	-3.0969e-004	-2.4775e-004	-6.1938e-005	2.0000e+001
y_{14}	1.6759e-005	1.3407e-005	3.3518e-006	2.0000e+001
y_{20}	2.2145e-003	2.6574e-003	-4.4290e-004	-2.0000e+001
y_{21}	4.1878e-004	3.3502e-004	8.3756e-005	2.0000e+001
y_{22}	1.3117e-005	1.2985e-005	1.3152e-007	1.0026e+000
y_{23}	-1.1549e-006	-9.2392e-007	-2.3098e-007	2.0000e+001
y_{24}	-5.2196e-005	-5.6434e-005	4.2377e-006	-8.1188e+000
y_{25}	8.8682e-006	1.0258e-005	-1.3896e-006	-1.5670e+001
y_{26}	-3.2717e-007	-3.3092e-007	3.7518e-009	-1.1467e+000
y_{30}	-1.6884e-003	-1.3864e-003	-3.0204e-004	1.7889e+001
y_{31}	-1.3637e-005	-1.4191e-005	5.5363e-007	-4.0598e+000
y_{32}	1.3214e-006	1.5857e-006	-2.6428e-007	-2.0000e+001
y_{40}	-1.4504e-001	-1.3696e-001	-8.0802e-003	5.5710e+000
y_{41}	1.3516e-002	1.3454e-002	6.2322e-005	4.6109e-001
y_{50}	1.3784e-001	1.3572e-001	2.1201e-003	1.5381e+000
y_{51}	3.5514e-003	3.0254e-003	5.2604e-004	1.4812e+001
l_{10}	2.2856e-003	2.6890e-003	-4.0336e-004	-1.7648e+001
l_{11}	6.4827e-005	6.0548e-005	4.2795e-006	6.6014e+000
l_{12}	-3.0529e-006	-2.5750e-006	-4.7789e-007	1.5654e+001
l_{13}	-2.7687e-005	-3.0562e-005	2.8752e-006	-1.0385e+001
l_{14}	1.7713e-006	2.1256e-006	-3.5426e-007	-2.0000e+001
l_{20}	2.9091e-003	2.3273e-003	5.8182e-004	2.0000e+001
l_{21}	9.0047e-006	1.0806e-005	-1.8009e-006	-2.0000e+001
l_{22}	-7.4562e-006	-5.9650e-006	-1.4912e-006	2.0000e+001
l_{23}	3.0423e-007	2.4338e-007	6.0846e-008	2.0000e+001

Continued on next page

Table 5.1 – continued from previous page

Parameter	Nominal Value	Perturbed Value	Δ Value	Δ in (%)
l_{24}	-2.5531e-005	-2.5053e-005	-4.7763e-007	1.8708e+000
l_{25}	4.1263e-006	4.5493e-006	-4.2304e-007	-1.0252e+001
l_{26}	-2.0918e-007	-2.5102e-007	4.1836e-008	-2.0000e+001
l_{30}	-4.4336e-001	-3.9473e-001	-4.8635e-002	1.0970e+001
l_{31}	7.5577e-004	7.0922e-004	4.6545e-005	6.1586e+000
l_{32}	-1.3921e-004	-1.1137e-004	-2.7842e-005	2.0000e+001
l_{40}	7.6582e-002	8.4423e-002	-7.8411e-003	-1.0239e+001
l_{41}	1.0019e-002	8.0152e-003	2.0038e-003	2.0000e+001
l_{42}	1.1783e-005	9.7436e-006	2.0394e-006	1.7308e+001
n_{10}	-1.5474e-003	-1.2379e-003	-3.0948e-004	2.0000e+001
n_{11}	6.1309e-005	7.1047e-005	-9.7381e-006	-1.5884e+001
n_{12}	-1.8989e-006	-2.2787e-006	3.7978e-007	-2.0000e+001
n_{13}	-5.5706e-006	-5.8670e-006	2.9642e-007	-5.3212e+000
n_{20}	7.7238e-004	8.6989e-004	-9.7508e-005	-1.2624e+001
n_{21}	1.1379e-006	1.3655e-006	-2.2758e-007	-2.0000e+001
n_{22}	-4.1705e-008	-3.3364e-008	-8.3410e-009	2.0000e+001
n_{30}	-1.5512e-002	-1.2433e-002	-3.0789e-003	1.9849e+001
n_{31}	-1.1325e-002	-1.2998e-002	1.6726e-003	-1.4769e+001
n_{32}	9.8251e-005	1.1790e-004	-1.9650e-005	-2.0000e+001
n_{40}	-8.5307e-002	-7.1651e-002	-1.3656e-002	1.6008e+001
n_{41}	8.0338e-004	8.6806e-004	-6.4677e-005	-8.0506e+000
n_{42}	-2.6197e-004	-2.0958e-004	-5.2394e-005	2.0000e+001
I_{xx}	5.0620e-001	4.0496e-001	1.0124e-001	2.0000e+001
I_{yy}	8.9000e-001	1.0080e+000	-1.1799e-001	-1.3258e+001
I_{zz}	9.1000e-001	9.8239e-001	-7.2388e-002	-7.9547e+000
I_{xz}	1.5000e-003	1.5093e-003	-9.2803e-006	-6.1869e-001

Correspond to the Fig. 5.7, the Table 5.1, represents the percentage of perturbation given to all the parameters constituting aerodynamic derivative and moment of inertia terms. It

represents the given case of close loop NA design for the actual plant model with nonzero weights initialization obtained from preflight maneuver.

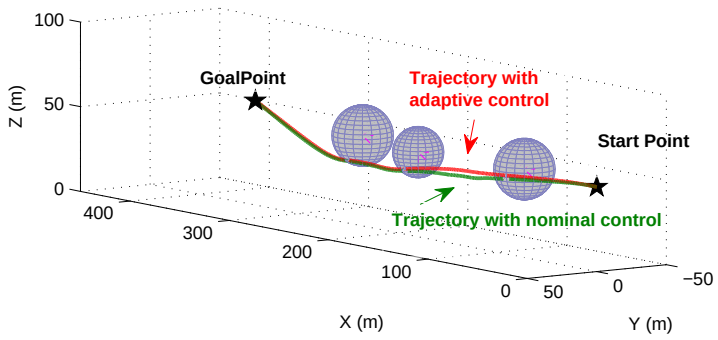


Figure 5.8: 3D scenario of obstacle avoidance with NA control

Figure 5.8 shows the $3D$ scenario with two trajectories corresponding to the nominal control applied on nominal plant and adaptive control applied to the actual plant. It can be seen that the adaptive control with actual plant is able to avoid the obstacle same as the nominal control with nominal plant. Figure 5.9 and Fig. 5.10 shows the longitudinal and lateral control deflections with three different profiles - nominal control with nominal states, adaptive control with actual states and the nominal control with actual states. It can be seen that the nominal control elevator deflection in the Fig. 5.9 hits the saturation limit of -25 deg when applied to the actual plant. Even in the Fig. 5.10 the nominal rudder deflection when applied to the actual states hits the saturation limit of 15 deg. However, the adaptive control applied to the actual plant does not saturate in none of the plots given by Fig. 5.9 and Fig. 5.10. Moreover, the adaptive control in Fig. 5.9 and Fig. 5.10 follows the same trend as that of the nominal control profile.

Figure 5.11 shows the tracking of the commanded values P^*, Q^*, R^* by the actual output of the inner loop with the adaptive control and the nominal control. It can be seen that the actual output with the nominal control fails to track the commanded values as compared to the actual output with the adaptive control. This shows the capability of the NA design in capturing the uncertainties of the actual plant. Figures 5.12, 5.13 and 5.14 shows the

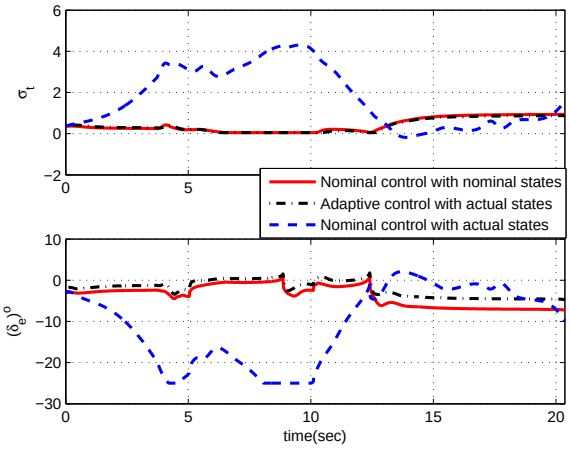


Figure 5.9: Longitudinal control deflections

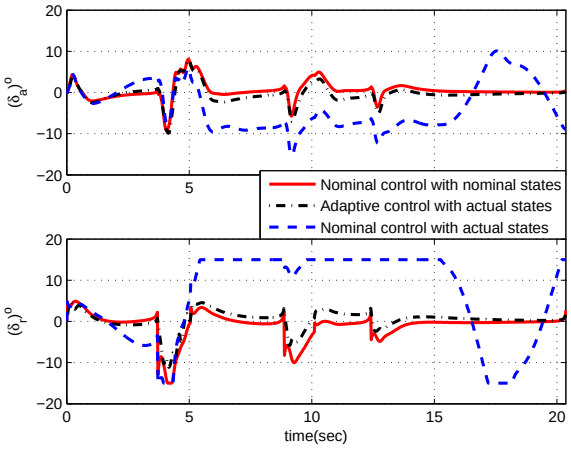


Figure 5.10: Lateral control deflections

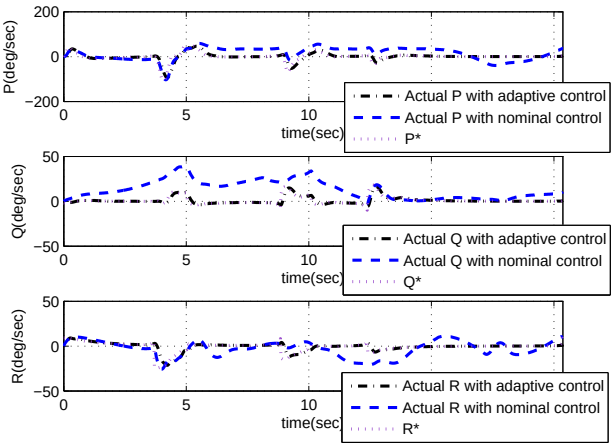


Figure 5.11: Body angular rates tracking

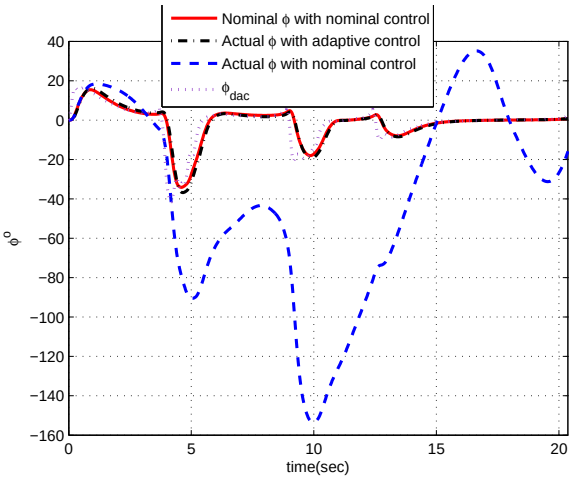


Figure 5.12: Tracking of guidance command ϕ_{dac}

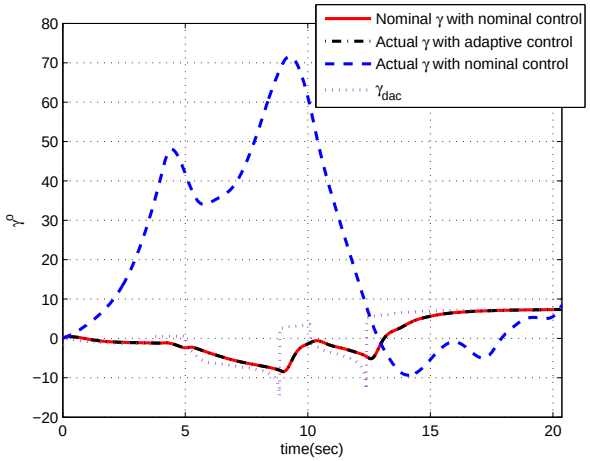


Figure 5.13: Tracking of guidance command γ_{dac}

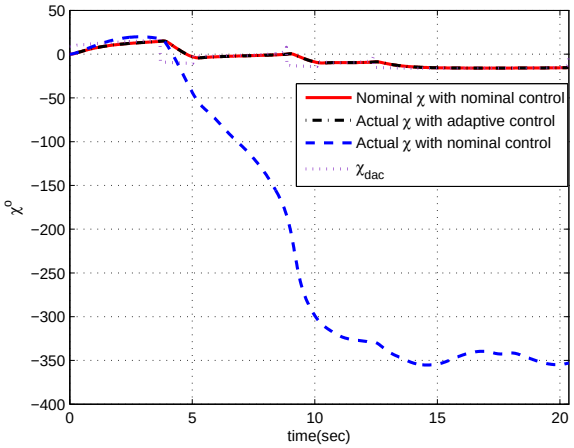


Figure 5.14: Tracking of guidance command χ_{dac}

tracking of the guidance commands ϕ_{dac}, γ_{dac}, χ_{dac} by the three comparative profiles. It can be seen that the profile representing the actual plant with adaptive control follows the nominal state profile effectively. Both the nominal state profile and the actual state profile with adaptive control tracks the guidance commands in contrast to the actual state with the nominal control. The actual ϕ and γ with the nominal control are only bounded as shown in Fig. 5.12 and Fig. 5.13, in contrast to the actual χ in Fig. 5.14 which is growing unboundedly. This implies that the nominal control fails to provide the robustness against uncertainties of the actual plant.

Figure 5.15 shows the function approximation $\hat{d_Y}(X)$ of the $d_Y(X)$ through the selected basis function for the output channels of the inner loop. It can be seen that the approximation of $d_Y(X)$ is very well achieved. Figure 5.16 represents the zoomed plot of the relative distance of UAV from obstacles, which shows that in case of all the obstacles, the safety ball incursion is within the specified limit i.e half of the wing span.

Table 5.2 shows the robustness study executed over the actual plant with the nominal control in comparison with the adaptive control. Three obstacles with different safety ball size around them in the environment is considered as the present case study for the robustness check. Different sets of the random perturbation both in inertia and coefficient terms are considered. It can be seen from the Table 5.2 that the actual plant sensitivity decreases drastically to the random perturbation in inertia and coefficient parameters, if it exceeds more than $\pm 20\%$. The actual plant is more sensitive to the aerodynamic coefficients than

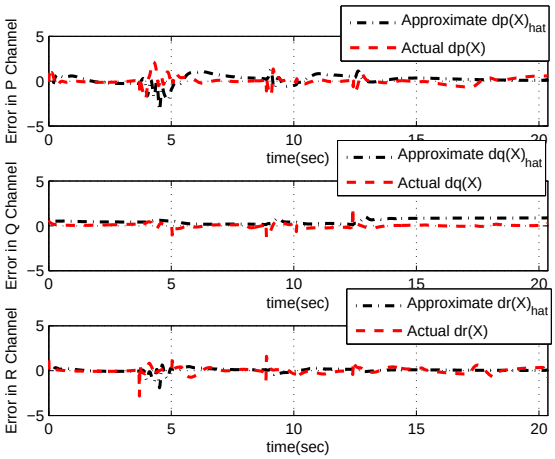


Figure 5.15: Capture of $d_Y(X)$ with $\hat{d_Y}(X)$

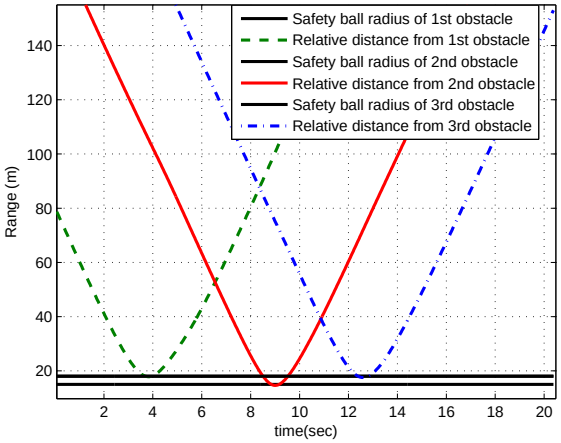


Figure 5.16: Relative distance of UAV from obstacles

Table 5.2: Robustness validation of actual plant with nominal control

Cases	Perturbation in Inertia terms	Perturbation in Aerodynamic Coefficients	Actual plant with nominal control	Actual plant with adaptive control
1	2%	1%	99%	100%
2	4%	2%	75%	100%
3	6%	3%	56%	100%
4	8%	4%	40%	100%
5	4%	4%	39%	100%
6	10%	5%	36%	100%
7	6%	6%	32%	100%
8	8%	8%	22%	100%
9	10%	10%	21%	100%
10	15%	15%	15%	98%
11	20%	20%	10%	82%

Table 5.3: Finite Vs Zero weights initialization for different scenarios

Scenario	NA design (zero weights)		NA design (finite weights)	
	Open loop actuator	Closed loop actuator	Open loop actuator	Closed loop actuator
1 obstacle	73%	77%	88%	94%
2 obstacles	77%	82%	88%	93%
3 obstacles	93%	98%	99%	100%

the inertia parameters. The success criteria used for validation of the plant robustness is as discussed before in chapter [4] for nominal PIGC design. One of the criteria is that the UAV incursion into the safety ball should be less than the half of the length of the wingspan i.e $1m$ and the other is the UAV should reach the goal point within the tolerance of $\pm 0.5m$. It can be inferred from the Table 5.2 that the actual plant with the adaptive control is more robust to the uncertainty of the plant than the nominal control which effectively proves the need of reinforcing the nominal PIGC design with the modified NA design. Table 5.3 shows the simulation study carried out for different scenarios with different number of obstacles with different safety ball size around them. It shows the behavior of the NA augmented PIGC design with the closed loop actuator and open loop actuator. In the case of closed loop NA design, comparison is shown with zero weights and finite weights initialization obtained from pre-flight maneuver. The success condition defined for the present study is same as that considered for the robustness study. It can be seen that from the Table 5.3 that the finite weights initialization works better than the zero weight initialization. It can also be inferred that from the Table 5.3, that the closed loop actuator performs better than the open loop first order actuator model. In case of closed loop actuator, it has its own controller which overcomes the delay introduced by the actuator dynamics. Moreover, as the number of obstacles keeps on increasing, the success rate also increases.

Chapter 6

Conclusions

UAVs are playing a vital role in numerous applications. Even in areas which are inaccessible to human beings, UAVs can outperform. The present work focuses on the reactive obstacle avoidance problem for unaccountable obstacles like urban edifices, poles etc. Unlike existing literature that mainly propose avoidance maneuvers using kinematic and point mass models, an innovative Six-DOF model based partial integrated guidance and control (PIGC) approach is presented for the obstacle avoidance.

Reactive maneuvers for obstacle avoidance demands guidance and control to execute in synergy in the IGC framework for fast corrections. However, due to the inherent separation between the guidance and control dynamics formed from the Six-DOF model, the IGC approach leads to unstable behavior of the system. On the contrary, PIGC performs the avoidance maneuver in the cascaded two loop structure which overcomes the shortcomings of the IGC approach, moreover, it reduces the delay in multiple loop tracking unlike the conventional design which may result fatal for the system. PIGC uses the guidance philosophy which is also validated with the point mass model of UAV as a test case. Test cases with the point mass model with a coordinated flight is demonstrated with all scenarios, where the controller dynamics were approximated as the first order autopilots.

The guidance strategy used in the PIGC approach uses the collision cone approach for obstacle detection and executes the avoidance maneuver by generating the angular guidance commands in the horizontal and the vertical planes. In the outer loop (i.e the guidance loop) of the NDI based PIGC approach, UAV pursues the guidance commands by quickly aligning its velocity vector along the aiming point while enforcing the turn coordination. With the

enforcement of the angular correction, the outer loop essentially generates the commanded body angular rates for the inner loop. In the inner loop (i.e the control loop), tracking of the commanded body rates is executed in order to generate the necessary control surface deflections. Since PIGC implemented with Six-DOF model uses NDI technique which gives a closed form solution to the controller therefore it is computationally inexpensive and can be implemented on onboard micro computers of UAVs. Simulations of PIGC design is executed with the first order actuator model. A controller for the first order actuator model is also proposed to reduce the actuator delay. Scenarios with different number and safety ball size of the obstacles have been considered along with UAV initial state perturbation for large number of simulations. In all the simulations, the PIGC design successfully has met the two success criteria of safety ball incursion and the goal point achievement within the tolerance limits.

To overcome the issues like modeling errors and parameter inaccuracies due to unsteady aerodynamics of the NDI technique used in PIGC, inner loop of PIGC is reinforced with neuro adaptive design. In the NA design, the neural network weight update rule provides online training of the weights. To enhance fast and stable training of the weights, preflight maneuvers are proposed. Preflight maneuvers provides stabilized pre-trained weights which prevents any misapprehensions in the actual avoidance problem. The closed form of NA design is implemented which reduces delay in tracking and invokes the guidance loop of PIGC through actual state feedback. This leads to faster adaptation and also helps in stabilizing the unstable plant quicker, thus adding robustness to the plant as a whole. The success of all the simulations in closed loop NA design also depends on the two success criteria of safety ball incursion and the goal point achievement within the tolerance limits. A comparative study was executed to observe the difference in the performance of the NA design with zero weight initialization and finite stabilized weight initialization. It was observed that the finite weights outperforms the zero weights initialization. In case of NA augmented PIGC approach, actuator controller was used for the adaptive control surface deflections which outshines the open loop actuator in the comparative study. The robustness study for large number of simulations has been carried out by randomly perturbing the coefficients and the inertia terms. This study clearly shows that the NA augmented PIGC design is more robust to the parameter perturbations compared to the nominal control when applied to the perturbed plant model. In all the simulations, all the constraints posed by the vehicle

capability are very well met within the available time-to-go.

The utility of PIGC design can be enhanced by augmenting it with features like introducing moving obstacles along with the stationary obstacles. Instead of spherical safety zones around the obstacles, more optimized shape of the safety zone like cylinder can be considered to accommodate obstacles like electric poles and wires. The obstacle position can be considered to be partially known with some noise and hence can be estimated from the Kalman filter. The obstacle information can also be processed through real passive sensors like cameras which may involve wide exploration in the computer vision techniques.

Acknowledgements

This work was supported by AOARD/AFRL, USA under the contract number $FA-23861014014$, which was being operated at Indian Institute of Science, Bangalore with the project code $SID/PC99188$.

Appendix A

Generic Theory of Control Design Techniques Used

To execute the geometric guidance law for obstacle avoidance, the control is needed which will be observed in terms of control required to steer the UAV with respect to the guidance command. Basic controller used for point mass model and Six-DOF model of real UAV is modeled through feedback linearization method [24] based nonlinear dynamic inversion (NDI). Feedback linearization is an approach to nonlinear control design that has attracted lots of research in recent years. The central idea is to algebraically transform nonlinear systems dynamics into linear system through transformation and feedback. It differs entirely from conventional (Jacobian) linearization, where linear approximations of the dynamics is achieved about an operating point. However, as the NDI is rather highly sensitive to the issue of parameter inaccuracy and modeling errors, there is a strong need of augmenting this technique with some other robust/adaptive techniques, to make it useful in practice. A potential approach in this regard is the idea of online dynamic function approximation taking the help of evolving methods like 'neuro-adaptive technique'[28]. The main philosophy underlying the neuro-adaptive technique is that the neural networks have the universal function approximation property, which helps a controller to adapt the uncertainties of the plants.

A.1 Nonlinear Dynamic Inversion Design

A popular technique, which serves as a 'universal gain scheduling controller' (and hence avoids the tedious gain scheduling process), is nonlinear dynamic inversion (NDI) [23]. This technique is essentially based on the technique of feedback linearization [24]. It leads to a number of potential advantages; namely asymptotic (rather exponential) stability of the error dynamics thereby leading to perfect tracking, a simple closed form expression for the controller (hence no computational concerns), preserving the benefits of the PID design etc.

In the present work, the guidance and control design is formulated in two loop structure. Time scale/cascaded NDI technique is used which follows with the assumption of inner loop being faster than the outer loop. The nonlinearities in NDI, are canceled by output feedback. This is achieved by enforcing stable error dynamics so that error goes to zero with appreciable tracking [25]. Well known NDI technique is discussed in generic frame in this section. We focus on a class of nonlinear and control affine systems which can be represented by the following system dynamics:

$$\dot{X} = f(X, U) \tag{A.1}$$

This can be rewritten in the standard control-affine form:

$$\dot{X} = f(X) + g(X)U \tag{A.2}$$

$$Y = h(X) \tag{A.3}$$

where $X \in \mathbf{R}^n$, $U \in \mathbf{R}^m$ $Y \in \mathbf{R}^p$, are the state, control and performance output vectors of the nominal system respectively. The system is assumed to be point-wise controllable. The objective is to design a controller U so that $Y \to Y^*$ as $t \to \infty$, where Y^* is the commanded signal for Y to track. It is assumed that $Y^*(t)$ is bounded, smooth and slowly-varying. To achieve the above objective, we notice that from Equations (A.2) and (A.3), using the chain rule of differentiation, the expression for the first order derivative of Y can be written as

$$\dot{Y} = f_Y(X) + g_Y(X)U \tag{A.4}$$

where

$$f_Y(X) \triangleq \left[\frac{\partial h}{\partial X}\right] f(X) \tag{A.5}$$

$$g_Y(X) \triangleq \left[\frac{\partial h}{\partial X}\right] g(X) \tag{A.6}$$

Here, the *"relative degree"* of the system is one. Note that, the relative degree depends on the selection of outputs, as well as on the system dynamics. Next, defining the error $E \triangleq Y(t) - Y^*(t)$ the controller is synthesized such that the following stable first order error dynamics is satisfied:

$$\dot{E} + KE = 0 \tag{A.7}$$

where K is chosen to be positive definite gain matrix. A relatively easier way is to choose K as a diagonal matrix with positive elements in the diagonal. For better physical interpretation, one can choose the relevant parameters of the first-order system, we can choose $K = diag(1/\tau_1, \ldots, 1/\tau_n)$, where $\tau_i (i = 1, \ldots, m)$ represent 'time constant' of the i^{th} error channel. Next, using the definition of E and substituting the expression for $\dot{Y}$ from Eq. (A.4) in Eq.(A.7) with the necessary algebra, following expression is obtained.

$$g_Y(X)U = \beta \tag{A.8}$$

where

$$\beta = [-f_Y(X) + \dot{Y}^* - K(Y - Y^*)] \tag{A.9}$$

If $p = m$ (i.e., the system has same number of outputs as number of inputs) and $g_Y(X)$ is non-singular $\forall\, t$, then from Eq.(A.8) one can obtain the control solution as

$$U = [g_Y(X)]^{-1}\beta \tag{A.10}$$

This completes an overview of the basic steps of dynamic inversion control design. However, we wish to mention some salient features of this technique. First, note that it leads to a closed-form solution for the controller, and hence it can be implemented online without any computational difficulties. Moreover, the control solution in Eq.(A.10) ensures that $E \to 0$ as $t \to \infty$ 'asymptotically'(rather than exponentially). In other words, asymptotic tracking is achieved. Interested readers can find more details about dynamics inversion approach in references [23] and [25].

A.2 Neuro - Adaptive Design

Albeit, the NDI technique has evolved as a promising tool for nonlinear control design substituting the extensive gain scheduling approach, there are a few critical issues with respect

to the technique as well. One of the common issues are modeling errors and parameter inaccuracies due to unsteady aerodynamics. It leads to partial cancelation of the nonlinearities due to inversion of the model which makes the technique sensitive to the parameter uncertainties and hence, there is a need to augment this technique with adaptive/robust control design tools. Aerodynamic and inertia parameter inaccuracies are addressed by reinforcing the NDI technique with a neuro-adaptive design approach [28]-[30]. The basic philosophy of neuro-adaptive control design is carried out in two steps: (i) synthesis of a set of neural networks which capture matched unmodelled (neglected) dynamics or model uncertainties because of parametric variations and (ii) synthesis of a controller that drives the state of the actual plant to that of a desired nominal model. The neural network weight update rule is derived using Lyapunov theory [24], which guarantees both stability of the error dynamics (in a practical stability sense) and boundedness of the weights of the neural networks [26]. Note that, albeit this technique has been used with NDI controller, its procedure is independent of the nominal control design technique. Hence, it can be used in conjunction with any known technique which is used to design the nominal controller. Note that the proposed approach is particularly concerned about the output robustness (i.e. performance robustness) [31]. However, the necessary steps for the execution of neuro-adaptive control are given in the subsequent section.

In this section, we present a neuro-adaptive control design approach, which is capable of addressing the issue of parameter inaccuracy in the model. The philosophy of the approach lies in the fact that the difference of parameter values from their nominal values essentially generate unknown algebraic terms in the model. These unknown functions are captured by neural networks, which are trained online. In this approach, the first aim is to come up with a nominal controller, which will meet the goals for the nominal model. The class of nonlinear system which is focused can be represented by the following equation

$$\dot{X}_d = f_d(X_d) + G_d(X_d)U_d \tag{A.11}$$

$$Y_d = h_d(X_d) \tag{A.12}$$

where $X_d \in \mathbb{R}^n$ and $U_d \in \mathbb{R}^m$, $Y_d \in \mathbb{R}^p$ are the state, output and control variables of the nominal system respectively. The objective here is to design a controller U_d so that $Y_d \rightarrow Y^*$, where $Y_d(t)$ is the commanded signal, which is assumed to be bounded and smooth. The nominal controller U_d has been designed using NDI technique [32]. Equation (A.13) may not

truly represent the actual plant because of the presence of the uncertainties in the model. Using the chain rule of derivative,the expression for $\dot{Y}$ can be derived as:

$$\dot{Y}_d = f_{Y_d}(X_d) + G_{Y_d}(X_d)U_d \tag{A.13}$$

where $f_{Y_d} \triangleq [\partial h/\partial X_d] f_d(X_d)$ & $G_{Y_d} \triangleq [\partial h/\partial X_d] G_d(X_d)$. Now the actual plant output is represented as

$$\dot{Y} = f_Y(X) + G_Y(X)U + d_Y(X) \tag{A.14}$$

$d_Y(X)$ is an unknown function that arises due to parameter uncertainties and modeling errors. The controller U needs to be designed online such that the states of the actual plant follow the respective states of the nominal model. In other words, the goal is to ensure that $Y \to Y_d$ as $t \to \infty$. To achieve this, the idea followed here is to first capture the unknown function $d_Y(X)$, which is accomplished through a neural network approximation [26]. For this purpose, an intermediate step is needed, which is to define an *approximate system* as follows

$$\begin{aligned} \dot{Y}_a &= f_{Y_d}(X) + G_{Y_d}(X)U + \hat{d}_Y(X) + K_a(Y - Y_a) \\ Y_a(0) &= Y(0) \end{aligned} \tag{A.15}$$

where $f_{Y_d} \triangleq [\partial h/\partial X] f_d(X)$ & $G_{Y_d} \triangleq [\partial h/\partial X] G_d(X)$ and K_a is selected as a positive definite gain matrix. A relatively easy way of doing this is to select K_a as a diagonal matrix with the i^{th} element being $k_{a_i} > 0$. Even though the selection of $k_{a_i} > 0 \ \forall \ i = 1, \ldots, n$ satisfies the need of the K_a being positive definite matrix, it is desirable to choose $k_{a_i} > 0.5$ because it leads to a smaller bound in the tracking error (this will become clear towards the end of this section). Note that whenever a function is approximated and only the approximate function is kept in the dynamics, then the modified equation no more represents the true dynamics (because of the function approximation error). This is the primary reason to introduce the Y_a dynamics. The approach followed here for ensuring $Y \to Y_d$ involves two steps: (i) $Y \to Y_a$ and (ii) $Y_a \to Y_d$, which are discussed next. A pictorial representation of these steps is shown in the Fig. A.1.

Step1 Capturing $d_Y(X)$ and ensuring $Y \to Y_a$

To capture the unknown function, first write $d_Y(X) \triangleq \begin{bmatrix} d_{y_1}(X) & \ldots & d_{y_n}(X) \end{bmatrix}$ where $d_{y_i}(X), i = 1, 2, \ldots . n$ is the ith component of the $d_Y(X)$. Each $d_Y(X)$ is approximated as $\hat{d_Y}(X)$ in a separate linear-in-the-weight neural network. We assume that the unknown function $d_Y(X)$ can be represented in terms of the basis function vector $\phi(X)$.

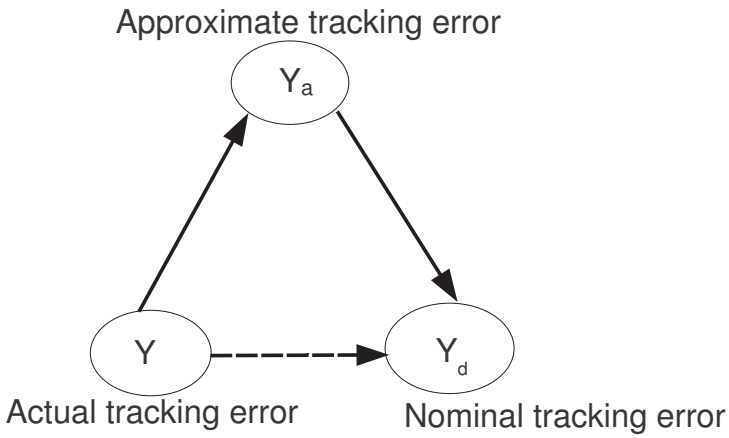


Figure A.1: Philosophy of model following approach

$$\hat{d_{y_i}}(X) = \hat{W_i}^T \phi_i(X) \tag{A.16}$$

where $\hat{W}_i$ is the weight vector of the i^{th} neural network and $\phi_i(X)$ is its basis function vector for each channel. This neural network function approximation is depicted in Fig. A.2. At

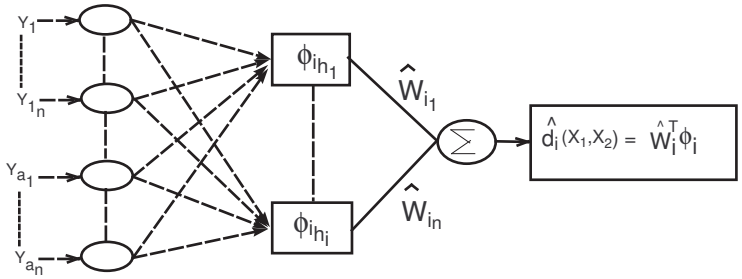


Figure A.2: Linear-in-weight neural network

this point, it needs to be mentioned that even though generic radial basis functions can be used for this purpose [30], [31],[32]. It is probably wiser to incorporate some prior knowledge about the system and judiciously select the basis functions, which will lead to faster learning of the unknown function. Note that the combination of n sub-networks can be interpreted to constitute a single neural network that represents $d_Y(X)$. The idea of having n neural

networks for n independent channels is to facilitate simpler mathematical analysis. More important, it leads to faster training because of reduced computational complexity, as none of the weights are linked to more than one output function. The next task is to update the weights of the neural network (i.e. to train them). Towards this end, the error between the actual state and the corresponding approximate state is defined as $e_{ai} \triangleq y_i - y_{a_i}$. The derivative of the error $\dot{e}_{a_i}$ in each channel can be given as

$$\dot{e}_{a_i} \triangleq \dot{y}_i - \dot{y}_{ai} \tag{A.17}$$

By substituting Eq. (A.14) and Eq. (A.15) in Eq. (A.17) the equations for the i^{th} channel $\dot{e}_{a_i}$ is written as

$$\begin{aligned} \dot{e}_{a_i} &\triangleq d_{y_i}(X) - \hat{d}_{y_i}(X) - k_{a_i} e_{ai} \\ &= \tilde{W_i}^T \phi_i(X) + \epsilon_i - k_{a_i} e_{ai} \end{aligned} \tag{A.18}$$

where $\tilde{W} \triangleq (W_i - \hat{W})$ is the error between the ideal weight and actual weight of the neural network. ϵ_i is the approximation error of the unknown function $d_{y_i}(X)$ being approximated. Next, define a series of Lyapunov function candidates $L_i, i = 1, 2, ...n$ such that

$$L_i = \frac{e_{ai} p_i e_{ai}}{2} + \frac{\tilde{W}_i^T {\gamma_i}^{-1} \tilde{W}_i}{2} \tag{A.19}$$

where $p_i > 0$ and $\gamma_i > 0$. Taking the time derivative of both sides of Eq. (A.19), and using the fact that $\dot{\tilde{W}} = -\dot{\hat{W}}_i$ (since W_i is constant) substitute $\dot{e_{ai}}$ from Eq. (A.18)

$$\begin{aligned} \dot{L}_i &= e_{ai} p_i \dot{e}_{a_i} + \tilde{W}_i^T {\gamma_i}^{-1} \dot{\tilde{W}}_i \\ &= e_{ai} p_i (\tilde{W}_i^T \phi_i(X) + \epsilon_i - k_{a_i} e_{ai}) + \tilde{W}_i^T {\gamma_i}^{-1} \dot{\tilde{W}}_i \end{aligned} \tag{A.20}$$

Note that our objective is to come up with a meaningful condition that will ensure $\dot{L}_i < 0$ which will ensure the stability of the error dynamics (of tracking error as well as weight error). However, the expression for $\dot{L}_i$ contains $\tilde{W}_i$ (which is unknown), and hence, nothing can be concluded about the sign of $\dot{L}_i$. To get rid of this difficulty, force the term multiplying it to zero and obtain the following weight update rule (training algorithm) for the i^{th} neural network.

$$\dot{\hat{W}}_i = \gamma_i e_{ai} p_i \phi_i(X) - \sigma_i \gamma_i \hat{W}_i \tag{A.21}$$

where γ_i can be interpreted as a learning rate for the i^{th} network (its numerical value essentially dictates the rate of capturing the unknown function $d_{y_i}(X)$). Note that Eq. (A.21) is the weight update (learning) rule for $\tilde{W}_i$. Select the initial condition as $\tilde{W}_i(0) = 0$. This

is compatible with the fact that if $d_{y_i}(X) = 0$ (i.e. there is no error in the model), then automatically $\hat{d_{y_i}}(X) = 0$. From the previous discussion we know that $\dot{\tilde{W}}_i = -\dot{\hat{W}}_i$, therefore Eq. (A.20) becomes

$$\dot{L}_i = e_{ai} p_i (\tilde{W}_i^T \phi_i(X) + \epsilon_i - k_{ai} e_{ai}) - \tilde{W}_i^T {\gamma_i}^{-1} \dot{\hat{W}}_i \tag{A.22}$$

Substituting $\dot{e_{a_i}}$ from Eq. (A.18) and $\dot{\hat{W}}_i$ from Eq. (A.21) in Eq. (A.22), we get

$$\dot{L}_i = e_{a_i} p_i \epsilon_i - k_{a_i} e_{a_i}^2 p_i + \sigma_i \tilde{W}_i^T \hat{W}_i \tag{A.23}$$

However $\tilde{W}_i^T \hat{W}_i$, the last term from Eq.(A.23) can be derived as follows

$$\tilde{W}_i^T \hat{W}_i = \frac{1}{2} \left(2 \tilde{W}_i^T \left(W_i - \tilde{W}_i \right) \right) = \frac{1}{2} \left(2 \tilde{W}_i^T W_i - 2 \tilde{W}_i^T \tilde{W}_i \right) \tag{A.24}$$

Further expanding $2{\tilde{W}_i}^T W_i$, the first term from Eq.(A.24) becomes

$$\begin{aligned} 2\tilde{W}_i^T W_i &= \tilde{W}_i^T W_i + \tilde{W}_i^T W_i \\ &= \tilde{W}_i^T \left(\hat{W}_i + \tilde{W}_i \right) + \left(W_i - \hat{W}_i \right)^T W_i \\ &= \tilde{W}_i^T \hat{W}_i + \tilde{W}_i^T \tilde{W}_i + {W_i}^T W_i - \hat{W}_i^T W_i \\ &= \hat{W}_i^T \left(\tilde{W}_i - W_i \right) + \tilde{W}_i^T \tilde{W}_i + {W_i}^T W_i \\ &= -\hat{W}_i^T \hat{W}_i + \tilde{W}_i^T \tilde{W}_i + {W_i}^T W \end{aligned} \tag{A.25}$$

Using the Eq. (A.25), the Eq. (A.24) can be expressed as

$$\begin{aligned} \tilde{W}_i^T \hat{W}_i &= \tfrac{1}{2} \left(-\hat{W}_i^T \hat{W}_i + \tilde{W}_i^T \tilde{W}_i + {W_i}^T W_i - \tilde{W}_i^T \tilde{W}_i - \tilde{W}_i^T \tilde{W}_i \right) \\ &= \tfrac{1}{2} \left(-\hat{W}_i^T \hat{W}_i - \tilde{W}_i^T \tilde{W}_i + {W_i}^T W_i \right) \\ &\leq \tfrac{1}{2} \left(-\left\| \tilde{W}_i \right\|^2 - \left\| \hat{W}_i \right\|^2 + \|W_i\|^2 \right) \end{aligned} \tag{A.26}$$

Therefore the last term in Eq. (A.23) satisfies the following inequality

$$\sigma_i {\tilde{W}_i}^T \hat{W}_i \leq -\frac{1}{2} \sigma_i \|\tilde{W}_i\|^2 - \frac{1}{2} \sigma_i \|\hat{W}_i\|^2 + \frac{1}{2} \sigma_i \|W_i\|^2 \tag{A.27}$$

Equation (A.23) can now be rewritten as

$$\begin{aligned} \dot{L}_i &\leq e_{a_i} p_i \varepsilon_i - e_{a_i}^2 p_i k_{a_i} - \tfrac{1}{2} \sigma_i \|\tilde{W}_i\|^2 - \tfrac{1}{2} \sigma_i \|\hat{W}_i\|^2 + \tfrac{1}{2} \sigma_i \|W_i\|^2 \\ &\leq \tfrac{e_{a_i}^2 p_i}{2} + \tfrac{\varepsilon_i^2 p_i}{2} - e_{a_i}^2 p_i k_{a_i} - \tfrac{1}{2} \sigma_i \|\tilde{W}_i\|^2 - \tfrac{1}{2} \sigma_i \|\hat{W}_i\|^2 + \tfrac{1}{2} \sigma_i \|W_i\|^2 \\ &\leq \tfrac{e_{a_i}^2 p_i}{2} - e_{a_i}^2 p_i k_{a_i} + \left(\tfrac{\varepsilon_i^2 p_i}{2} - \tfrac{1}{2} \sigma_i \|\tilde{W}_i\|^2 - \tfrac{1}{2} \sigma_i \|\hat{W}_i\|^2 + \tfrac{1}{2} \sigma_i \|W_i\|^2 \right) \end{aligned} \tag{A.28}$$

In the expression (A.28), to achieve stability $\dot{L}_i < 0$. It is only possible if $(e_{a_i}^2 p_i k_{a_i} - \frac{e_{a_i}^2 p_i}{2}) > \beta_i$, where $\beta_i \triangleq \left[\frac{\varepsilon_i^2 p_i}{2} - \frac{1}{2}\sigma_i\|\tilde{W}_i\|^2 - \frac{1}{2}\sigma_i\|\hat{W}_i\|^2 + \frac{1}{2}\sigma_i\|W_i\|^2\right]$. This implies $(k_{a_i} - \frac{1}{2}) > (\frac{\beta_i}{e_{a_i}^2 p_i})$ is the condition for which $\dot{L}_i < 0$. It can be deduced that by selecting a small σ_i and sufficiently good set of basis functions, the approximation error ε_i will be reduced which will help in keeping the error bound small. Moreover, small ε_i leads to $\beta_i \geq 0$ which results in the condition that $k_{a_i} \geq 0.5$.

Step2 Ensuring $Y_a \rightarrow Y_d$ and computation of U

As pointed out earlier, while ensuring $Y \rightarrow Y_a$ and capturing the unknown function $d_Y(X)$ as a functional approximation $\hat{d_Y}(X)$, it is simultaneously ensured that $Y_a \rightarrow Y_d$ as $t \rightarrow \infty$. To achieve this objective, the controller U is designed such that the following stable error dynamics is satisfied

$$(\dot{Y}_a - \dot{Y}_d) + K_g(Y_a - Y_d) = 0 \tag{A.29}$$

where K_g is chosen to be a positive definite gain matrix. A relatively easy way of selecting the gain matrix is to consider $K_g = diag(1/\tau_1 \ldots 1/\tau_n)$, where τ_i can be interpreted as the desired time constant for the i^{th} channel of the error dynamics in Eq. (A.29). By substituting Eq. (A.13) and Eq. (A.15) in Eq. (A.29)we get

$$f_{Y_d}(X) + G_{Y_d}(X)U + \hat{d_Y}(X) + K_a(Y - Y_a) - f_{Y_d}(X_d) - G_{Y_d}(X_d)U_d + K_g(Y_a - Y_d) = 0 \tag{A.30}$$

Now, by carrying out necessary algebra, the adaptive control is obtained as

$$U = -[G_{Y_d}(X)]^{-1}(f_{Y_d}(X) + \hat{d_Y}(X) + K_a(Y - Y_a) - f_{Y_d}(X_d) - G_{Y_d}(X_d)U_d + K_g(Y_a - Y_d)) \tag{A.31}$$

The adaptive controller in Eq. (A.31) is designed based on NDI in the present work. Note that, the second step of the NA design is independent of the the controller technique used in designing the adaptive controller. Further details of the NA design can be found in [28]-[32].

Bibliography

[1] Beard, R., Kingston, D., Quigley, M., Snyder, D., Christiansen, R., Johnson, W., McLain, T., and Goodrich, M. A., "Autonomous Vehicle Technologies for Small Fixed-Wing UAVs", Journal of Aerospace Computing, Information, and Communication, Vol. 2, January 2005.

[2] Hrabar, S., Sukhatme, G. S., Corke, P., Usher, K., and Roberts, J., "Combined optic-flow and stereo-based navigation of urban canyons for a UAV", *In Proceedings of IEEE International Conference on Intelligent Robots and Systems, Alberta*, 2004, pages 3609 3615.

[3] LaValle, S. M., and Kuffner, J. J., "Rapidly-exploring random trees: Progress and prospects", *Algorithmic and Computational Robotics: New Directions*, 2001.

[4] Scherer, S., Singh, S., Chamberlain, L., and Elgersma, M., "Flying Fast and Low Among Obstacles: Methodology and Experiments", *The International Journal of Robotics Research*, 2008, Vol. 27, No. 5, pages 549 574.

[5] Shim, D. H., Chung, H., and Sastry, S., "Autonomous Exploration in Unknown Urban Environments for Unmanned Aerial Vehicles", *IEEE Robotics and Automation Magazine*, 2006, Vol. 13, No. 3, pages 27 - 33.

[6] Hwangbo, M., Kuffner, J., and Kanade, T., "Efficient Two-phase 3D Motion Planning for Small Fixed-wing UAVs", *IEEE International Conference on Robotics and Automation*, 10 - 14 April 2007, Rome, Italy.

[7] Han, S. C., and Bang, H., "Proportional Navigation-Based Collision Avoidance for UAVs", *International Journal of Control, Automation and Systems*, 2009, Vol.7, No.4, pages 553 - 565.

[8] Watanabe, Y., Calise, A. J., and Johnson, E. N., "Minimum Effort Guidance for Vision-Based Collision Avoidance", *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, 21 - 24 August 2006, Keystone, Colorado.

[9] Chakravarthy, A., and Ghose, D., "Obstacle Avoidance in a Dynamic Environment: A Collision Cone Approach", *IEEE Transactions on Systems, Man and Cybernetics-Part A: Systems and Humans*, 1998, Vol. 28, No. 1, pages 562 - 574.

[10] Carbone, C., Ciniglio, U., Corraro, F., and Luongo, S., "A Novel 3D Geometric Algorithm for Aircraft Autonomous Collision Avoidance", *Proceedings of the* 45^{th} *IEEE Conference on Decision and Control*, 13 - 15 December 2006, San Diego, CA, USA.

[11] Bilimoria, K. D., "A Geometric Optimization Approach to Aircraft Conflict Resolution", *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 14 - 17 August 2000, Denver, CO.

[12] Park, J. W., Oh, H. D., and Tahk, M. J., "UAV Collision Avoidance Based on Geometric Approach", *SICE Annual Conference*, 2008, pages 2122 - 2126.

[13] Hull, D. G., "Fundamentals of Airplane Flight Mechanics", *Springer*, 2007.

[14] Noa, T. S., Mina, B. M., Stoneb, R. H., and Wong K. C.,"Control and Simulation of Arbitrary Flight Trajectory-Tracking",*Control Engineering Practice*, 2005, Vol. 13, No. 5, pages 601 - 612.

[15] Mujumdar A., and Padhi, R.,"Nonlinear Geometric and Dierential Geometric Guidance of UAVs for Reactive Collision Avoidance",*AOARD Project, 2010, Project Ref. No. IISc/SID/AE/LINCGODS/AOARD/2010/01*,Department of Aerospace Engineering, Indian Institute of Science, Bangalore.

[16] Xin, M., Balakrishnan, S. N., Stansbery, D. T., and Ohlmeyer, E. J., "Nonlinear Missile Autopilot Design with $\theta - D$ Technique", *Journal of Guidance, Control, and Dynamics*, Vol.27, No.3, May-June 2004.

[17] Xin, M., Balakrishnan, S. N., and Ohlmeyer, E. J., "Integrated Guidance and Control of Missiles with $\theta - D$ Method", *Proceedings of* 16^{th} *IFAC Symposium on Automatic Control in Aerospace*, 14 - 16 June, 2004, St. Petersburg, Russia.

[18] Padhi, R., Chawla, C., Das, P. G., and Venkatesh, A., "Partial Integrated Guidance and Control of Surface-to-Air Interceptors for High Speed Targets", *American Control Conference*, 10-12 June 2009, St. Louis, USA.

[19] Chawla, C., and Padhi, R., "Reactive Obstacle Avoidance of UAVs with Dynamic Inversion Based Partial Integrated Guidance and Control", *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2 - 5 August 2010, Toronto, Canada.

[20] Stevens, B., and Lewis, F., "Aircraft Control and Simulation 2^{nd} Edition", *J.Wiley & Sons*, 2003.

[21] Shenoydor, N. A., "Missile Guidance and Pursuit: Kinematics, Dynamics and Control",*Horwood Publishing Limited*, 1998.

[22] Tsao, P. L., Chou, C. L., Chen, C. M., and Chen, C. T., "Aiming Point Guidance Law for Air-to-Air Missiles", *International Journal of Systems Science*, 1998, Vol. 29, No. 2, pages 95 - 102.

[23] Enns, D., Bugajski, D., Hendrick, R., and Stein, G., "Dynamic inversion: an evolving methodology for flight control design", *International Journal of Control*, 1994, Vol. 59, No. 1, pages 71 - 91.

[24] Slotine, J. J. E., and Li, W., "Applied Nonlinear Control", *Prentice Hall*, 1991.

[25] Padhi, R., and Balakrishnan, S. N., "Implementation of Pilot Commands in Aircraft Control: A New Approach Based on Dynamic Inversion", *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 11 - 14 August, 2003, Austin, Texas.

[26] Haykin, S., "Neural Networks: A comprehensive foundation", *Prentice Hall*, 2006.

[27] Wang, Q., and Stengel, R. F., "Robust nonlinear flight control of a high-performance aircraft", *IEEE Transactions on Control Systems Technology* , 2005, Vol. 13, No. 1, pages 15 - 26.

[28] Padhi, R., Unnikrishnan, N., and Balakrishnan, S. N., " Model Following Neuro-Adaptive Control Design for Non-square, Non-affine Nonlinear System", *IET Control Theory and Applications* , 2007, Vol.1, No. 6, pages 1650 - 1661.

[29] Padhi, R., and Kothari, M., "An optimal dynamic inversion based neuro-adaptive approach for treatment of chronic myelogenous leukemia", *Computer methods and Programms in Biomedicine*, 2007, Vol. 87, No. 3, pages 208 - 224.

[30] Rajasekharan, J., Chunodkar, A., and Padhi, R.,"Structured model-following neuro-adaptive design for attitude maneuver of rigid bodies",*Control engineering practice*, 2009, Vol. 17, No. 6, pages 676 - 689.

[31] Das, P. G., Chawla, C., and Padhi, R., "Robust Partial Integrated Guidance and Control of Interceptors in Terminal Phase", *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 10 - 13 August 2009, Chicago, Illinois.

[32] Padhi, R., Rao, N. P., Goyal, S., and Tripathi, A.,"A Model-following Neuro-Adaptive Approach for Robust Control of High Performance Aircrafts", *Automatic Control in Aerospace*, 2010, Vol. 3, No. 1.

[33] Singh, S. P., and Padhi, R., "Automatic Path Planning and Control Design for Autonomous Landing of UAVs using Dynamic Inversion", *American Control Conference*, 10 - 12 June 2009, St. Louis, USA.

[34] Singh, S. P., "Autonomous Landing of Unmanned Air Vehicles", Department of Aerospace Engineering, Indian Institute of Science, Bangalore, 2009.

[35] Surendra nath, V., Govindaraju, S. P., Bhat, M. S., and Rao, C. S. N., "Configuration Development of All Electric Mini Airplane", *ADE/DRDO Project, 2004, Project Ref. No: ADEO/MAE/VSU/001*, Department of Aerospace Engineering, Indian Institute of Science, Bangalore.

[36] Atkinson, K. E.,"An Introduction to Numerical Analysis",*John Wiley & Sons*, 2001.